DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA AÉREO DE CONTEO EN PLANTACIONES DE PALMA DE ACEITE

JHANY ZULYMA MISERQUE CASTILLO
RUBBERMAID LAVERDE DIAZ

UNIVERSIDAD PONTIFICIA BOLIVARIANA
ESCUELA DE INGENIERÍAS
FACULTAD DE INGENIERÍA ELECTRÓNICA
BUCARAMANGA
2015

DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA AÉREO DE CONTEO EN PLANTACIONES DE PALMA DE ACEITE

JHANY ZULYMA MISERQUE CASTILLO
RUBBERMAID LAVERDE DIAZ

Trabajo de grado para optar por el título de Ingeniero Electrónico

Director
MSc. Claudia Leonor Rueda Guzmán

UNIVERSIDAD PONTIFICIA BOLIVARIANA
ESCUELA DE INGENIERÍAS
FACULTAD DE INGENIERÍA ELECTRÓNICA
BUCARAMANGA
2015

Dedicatoria

A las personas que, junto a Dios, me han traído hasta aquí y jamás me han abandonado. Nunca han permitido que nada me falte. Me han apoyado y amado cada día sin importar los errores. A mis padres que han dado todo por mis sueños, quiero hoy dedicarles el fruto de mi esfuerzo.

Para mi abuela que está siempre pendiente de cada detalle de mi día a día para consentirme y cuidarme, en especial en todo este tiempo que yo le he dedicado mi energía a este proyecto. Renunciaste a tus charlas conmigo para que la prioridad fuese mi trabajo, hoy quiero entregarte lo que tu me ayudaste a lograr.

A mi hermana que siempre tiene algo gracioso, de lo que pocos se ríen, pero ella sabe que decirme para animarme en los días de trabajo. Para la única que tengo, para ti va un pedacito del trabajo con todo mi cariño.

A Rubbermaid, por entenderme con tanta paciencia incluso cuando no tenía razón ni sentido lo que decía. Porque siempre has apoyado mis proyectos y locuras. Este libro ya es tuyo, pero yo te dedico el tiempo y la energía que le di a este proyecto, en vez de dedicarla a nosotros.

A Helen y Luz Ángela, que probablemente no lean el contenido del libro pero entendieron que no podía jugar con ellas porque estaba haciendo mi trabajo, y sin embargo, junto a Luchy, siempre estuvieron pendientes de mí y encontraron un minuto para hacerme reír y divertirme. Para ustedes también es este libro.

Para mis profesores, los que les entendí todo y aprendí claramente, los que no les entendí nada y tuvieron que repetirme mil veces, los que respondieron tantas preguntas, los que creyeron en mí y los que primero dudaron, los que me dedicaron su tiempo para alimentar mi creatividad y mi mente, hoy les dedico todo lo que aprendí aquí.

Y varias ideas que desarrollé y páginas que escribí son para mis amigas, porque no respondí los mensajes ni estuve con ellas por dedicarle tiempo a esto, pero aún así siguen conmigo.

Porque no llegue hasta aquí sola, para ustedes que compartieron mi esfuerzo, hoy les comparto mi alegría. Gracias.

Jhany.

Dedicatoria

Dedico este proyecto especialmente a mis padres por brindarme su apoyo incondicional en todo momento, por no dejar de creer en mi aun en los momentos difíciles, y por apoyar todas mis ideas por más descabelladas que pudieran ser. A ellos, quienes con su esfuerzo y dedicación han hecho de mi la persona que hoy soy, con mucho amor e infinita gratitud, les dedico todo el tiempo y el esfuerzo invertido en este proyecto.

A Jhany, mi compañera de trabajo incondicional. Se que fueron muchos momentos difíciles, y aunque siempre parecía que estábamos locos por embarcarnos en los proyectos mas complejos, aceptando retos que ni siquiera nosotros mismos sabíamos que podíamos superar, hoy quiero decirte que todo ese esfuerzo valió pena y que sin ti no habría sido posible llegar hasta aquí, gracias por tu paciencia.

A mis hermanos y mi familia en general, por el interés constante en todos mis proyectos, aunque a veces se que no entienden nada de lo que hago.

A doña Ruby y a la señora Zuly les dedico especialmente este proyecto por estar tan pendientes de nosotros y preocuparse por mantenernos siempre bien alimentados, parecía que nos querían engordar para comernos en Navidad. Muchas gracias por el todo el apoyo y el banquete.

A Max, la bola de pelos de la casa, a él le dedico este proyecto y todas aquellas largas jornadas de trabajo en las que estuvo conmigo, aunque dormido y perdido en sus sueños caninos, siempre fue una compañía agradable en momentos de desesperación.

Finalmente, le dedico también este proyecto a mi Xbox One, por tener que soportar mi abandono y a pesar de ello entender muy bien cuáles son mis prioridades. Prometo regalarle muchos videojuegos como compensación.

A todos ellos, gracias. Esto es para ustedes.

Rubbermaid.

Agradecimientos

A la Universidad Pontificia Bolivariana por el apoyo dado para el desarrollo del proyecto de investigación y la obtención del UAV necesario para el trabajo en campo.

A la MSc. Claudia Rueda Guzmán por su guía constante para el desarrollo de la investigación, su entusiasmo por el proyecto y su voluntad inquebrantable para sacar lo mejor de nosotros. Gracias por todas las horas extras dedicadas a ayudarnos a solucionar problemas, a leer todo lo que enviábamos y a llenar formularios para el éxito administrativo de la investigación.

A los propietarios de AGROLAV por permitirnos realizar allí numerosas pruebas de vuelo para obtener las imágenes necesarias e incluso modificar sus tareas del día para permitirle a sus trabajadores compartir con nosotros su experiencia y guiarnos en la plantación.

Al señor Guillermo Mantilla y a la señora Elizabeth García de Palmares El Pórtico por permitirnos evaluar nuestro sistema en sus plantaciones y brindarnos una amable guía durante todo el tiempo de nuestra estadía.

Al director de la Facultad de Ingeniería Electrónica, el MSc. Raúl Restrepo por promover y apoyar el desarrollo de investigaciones basadas en todas las ideas que soñamos como ingenieros.

Al Dr. Miguel Altuve por compartirnos su invaluable conocimiento en el área de aprendizaje automático que nos abrió las puertas a técnicas que hicieron posible el éxito del proyecto.

Al Dr. Luis Ángel por los cimientos planteados en clase que impulsaron nuestra motivación para trabajar con imágenes y su voluntad constante de resolver cualquier duda que se presentara en nuestro camino.

Al Dr. Jhon Jairo Padilla por creer en los alcances del proyecto y presentarlo en RiegoNets, dándonos la oportunidad de ampliar las plantaciones de prueba y permitiendo que el proyecto se convierta en base para continuar con investigaciones futuras.

A Robinson, por recibirnos tan amablemente cada vez que íbamos a la plantación a probar los avances del proyecto y sacar de su tiempo de trabajo, e incluso de su descanso, para apoyar nuestras ideas y ayudarnos a solucionar los problemas.

Al Dr. Omar Pinzón por permitirnos aplazar los compromisos que teníamos pendientes con él para darle prioridad al proyecto.

Al Ing. Alex Monclou, por apoyar e incentivar todas nuestras ideas desde el primer día que llegamos a la universidad y siempre creer que podíamos hacer más.

A nuestras familias que nos apoyaron constantemente para que pudiéramos concentrarnos y darle prioridad a nuestro trabajo.

A Nelson y Gabriel por compartir con nosotros este año de esfuerzos y sacrificios y brindarnos su constante ayuda siempre que los necesitábamos.

TABLA DE CONTENIDO

pág.

LISTA DE FIGURAS

pág.

RESUMEN GENERAL DE TRABAJO DE GRADO

TITULO: Desarrollo e implementación de un sistema aéreo de conteo en plantaciones de palma de aceite

AUTOR(ES): Jhany Zulyma Miserque Castillo
Rubbermaid Laverde Diaz

FACULTAD: Facultad de Ingeniería Electrónica

DIRECTOR(A): Claudia Leonor Rueda Guzmán

RESUMEN

Este proyecto plantea el desarrollo e implementación de un sistema que permite capturar, procesar y analizar imágenes de una plantación de palma de aceite para registrar la cantidad de palmas cultivadas mediante el uso de un vehículo aéreo no tripulado (UAV), herramientas de fotogrametría y técnicas de visión artificial. Se inicia con un estudio de las tecnologías de UAVs disponibles para definir el modelo más adecuado a las necesidades del proyecto. Como resultado se adquiere un DJI Phantom 2 Vision+, el cual es controlado por medio de un dispositivo móvil donde se le establece la ruta de vuelo y los puntos en los cuales deben adquirirse las fotografías que luego serán procesadas mediante un software de fotogrametría para crear ortomosaicos de las áreas de interés. Posteriormente, estos ortomosaicos son procesados por el software de detección y conteo desarrollado en este proyecto para determinar el número de palmas que contienen. El algoritmo implementado utiliza una técnica de ventana deslizante con escalado piramidal para la generación de ventanas candidatas, un descriptor LBP (Local Binary Pattern) que modela matemáticamente la textura de la imagen, un modelo de regresión logística que opera como clasificador de ventanas y un algoritmo de non-maximum suppression para refinar la decisión. Dicho software se desarrolla en tres fases: entrenamiento, validación y evaluación; a través de las cuales se busca optimizar el sistema para obtener el mejor desempeño. Finalmente, al implementar el sistema en plantaciones reales se demuestra que los UAVs son una herramienta útil para obtener imágenes actualizadas con un bajo costo y sin problemas de nubosidad asociados a los satélites. Al integrar esta tecnología con el sistema de conteo desarrollado se obtienen resultados satisfactorios para la solución del problema, generando imágenes de alta calidad con información precisa de la zona de interés y un censo automatizado de la misma.

PALABRAS CLAVES:

Palma de aceite, UAV, fotogrametría, ortomosaico, visión artificial.

GENERAL SUMMARY OF WORK OF GRADE

TITLE:	Development and implementation of a counting aerial system in oil palm plantations
AUTHOR(S):	Jhany Zulyma Miserque Castillo Rubbermaid Laverde Diaz
FACULTY:	Facultad de Ingeniería Electrónica
DIRECTOR:	Claudia Leonor Rueda Guzmán

ABSTRACT

This project proposes the development and implementation of a counting aerial system capable to capture, process and analyze images of an oil palm plantation in order to register the amount of cultivated palms using an unmanned aerial vehicle (UAV), photogrammetry tools and computer vision techniques. It begins with a study of the UAV technologies available in the market to define the most appropriate model according to the project needs. As result, a DJI Phantom 2 Vision+ is acquired, which is controlled by a mobile device that sets the flight route and the points where the UAV must capture the pictures that will be processed by a photogrammetry software in order to create orthomosaics from the areas of interest. Later, these orthomosaics are processed by the detection and counting software developed in this project to calculate the number of palms contained in them. The implemented algorithm uses a sliding window technique in image pyramids to generate windows with possibility to contain a palm, a LBP descriptor (Local Binary Pattern) to mathematically model the texture of the image, a logistic regression model to classify the windows and a non-maximum suppression algorithm to refine the decision. This software is developed in three stages: training, validation and evaluation; through which the system is optimized to obtain the best performance. Finally, the implementation of the system in real plantations proved that the UAV is a useful tool to acquire updated images with a low cost and without the cloud problems associated to satellites. The integration of this technology with the developed counting system gave satisfactory results to solve the problem, getting the census and high-quality images with precise information from the area of interest.

KEYWORDS:

Oil palm, UAV, photogrammetry, orthomosaic, computer vision.

INTRODUCCIÓN

Colombia es el primer productor de aceite de palma en América y el cuarto a nivel mundial, por lo que el cultivo de palma de aceite es una actividad económica de gran importancia para el país. Estas plantaciones se caracterizan por abarcar grandes áreas que son afectadas por fenómenos naturales como plagas, enfermedades y caídas de rayos.

Las múltiples alteraciones hacen que el número de palmas pueda variar rápidamente y en esta industria, cada planta representa dinero, tiempo y servicios. Por tanto es indispensable mantener un control adecuado del censo y organización de las distintas zonas de la plantación.

Durante mucho tiempo este conteo se ha realizado de forma manual, con trabajadores que recorren los cultivos contando el número de palmas y anotando esta información con lápiz y papel. Esta técnica está sujeta a un alto nivel de error al depender completamente del factor humano que puede fallar especialmente si se tienen en cuenta las condiciones climáticas que representan estas plantaciones. Además, toma bastante tiempo realizar estos recorridos y anotaciones, por lo que no suele realizarse de forma periódica.

Para obtener información que pueda actualizarse más fácilmente y que cuente con un mayor nivel de confiabilidad, algunas empresas han recurrido a técnicas modernas como el empleo de satélites para obtener vistas aéreas de las plantaciones desde las cuales monitorear sus cultivos. Sin embargo, el alto nivel de nubosidad presente en los climas tropicales donde se siembra la palma, dificulta la captura de imágenes sin obstáculos; sin mencionar, que el acceso a estos recursos tiene un alto precio.

Ante esta problemática se plantea el uso de un vehículo aéreo no tripulado que sobrevuele la plantación para obtener imágenes de los cultivos que puedan procesarse para crear mapas u ortomosaicos de zonas de interés, a partir de los cuales se genere un censo de la población de palmas que no requiera un conteo por un factor humano.

Es así como este libro presenta el desarrollo e implementación de este sistema de conteo; iniciando con una descripción del problema y de las diferentes formas en que se han abarcado soluciones y propuesto desarrollos por investigaciones ya realizadas. Luego se describen las razones que justifican el desarrollo de este

sistema y los objetivos específicos que se desean cumplir. Adicionalmente, se realiza una breve introducción a la teoría requerida para comprender las herramientas utilizadas.

La segunda parte describe la estructura del sistema dividiéndolo en sus tres componentes principales: el análisis y selección del vehículo aéreo ideal para esta tarea, la creación de los ortomosaicos utilizando una herramienta de fotogrametría comercial y el desarrollo del software de detección y conteo de palmas en los mapas obtenidos. Allí mismo se desarrolla la evaluación del sistema que permite obtener una calificación de su rendimiento en plantaciones reales.

Finalmente, se presentan las conclusiones y recomendaciones futuras a partir de la experiencia obtenida con el sistema.

1. DEFINICIÓN DEL PROBLEMA

Las plantaciones de palma de aceite suelen caracterizarse por sus vastas extensiones que contienen un gran número de plantas, por lo que se hace difícil el diseño de los mapas del terreno. Hasta la fecha y en la mayoría de los casos, estos mapas son realizados por los cultivadores de forma manual y en largas jornadas de trabajo. De ahí que no se cuenta con información actualizada del número de palmas que se encuentran en buenas condiciones y que no han sido afectadas, o eliminadas, por eventos tales como caídas de rayos y enfermedades o plagas peligrosas. La gravedad de estas situaciones se incrementa cuando se considera que la mayoría de los procesos de mantenimiento realizados a cada palma se pagan de forma individual, de manera que la carencia de información se traduce en pérdidas económicas.

Ante esta problemática, se genera la necesidad de realizar un censo que pueda ser actualizado frecuentemente y con exactitud, de forma que proporcione la mayor cantidad posible de información acerca del contenido de la plantación, permitiendo que los cultivadores tengan un mayor control en la administración del tipo y cantidad de trabajo que debe realizarse, así como el manejo de los recursos físicos y económicos de la empresa.

2. ANTECEDENTES

La agricultura es una de las actividades económicas más antiguas en la historia de la humanidad, y a través de la experiencia se han conocido y detectado los problemas que inciden en su producción. En el caso de la vigilancia de la palma de aceite, Mariau[1] afirma que existen muchos enemigos que afectan el follaje y otros órganos de las palmas, incluyendo insectos, ácaros, mamíferos y aves; y que en contra de estas plagas es posible utilizar métodos de control, pero para obtener buenos resultados es preciso aplicar estas medidas oportunamente y para ello es necesario tener un buen conocimiento del estado fitosanitario de la planta. Esta necesidad de información de buena calidad obtenida preferiblemente en tiempo real, es ratificada por Araque y Forero[2], quienes resaltan que la oportunidad de la decisión es importante ya que el estado de las plantaciones cambia rápidamente en las condiciones del campo.

Para obtener la información requerida se han planteado varias técnicas de control entre las que se destacan las inspecciones de rutina descritas por el mismo Mariau[3] y que hasta el día de hoy, al revisar en contacto directo con los palmicultores, se siguen desarrollando de igual forma: con observaciones humanas realizadas por un trabajador al desplazarse a través de la plantación y revisar una muestra determinada por hectárea. Usando este mismo método se describe la elaboración de los mapas de defoliación que son especialmente útiles para el control de los daños y el desarrollo de las operaciones.

En 2014, Hoyos y Rincón[4] plantearon el uso de dispositivos móviles como herramienta auxiliar, de forma que a medida que el trabajador se desplaza por la plantación, registre la información en formularios electrónicos que luego serán descargados y procesados en un computador, obteniendo un mayor grado de confianza en los datos adquiridos y abriendo la posibilidad de visualizar la información en forma de tablas o mapas basados en los puntos GPS registrados en el recorrido.

[1] MARIAU, Dominique. Vigilancia sanitaria de las plantaciones de palma de aceite y cocotero. En: Palmas. 1995. vol 16, no. 1. p.25.
[2] ARAQUE, Leonardo y FORERO, Diana. Análisis de los sistemas de captura y procesamiento de información para la toma de decisiones en el manejo de los insectos defoliadores de la palma de aceite en Colombia. En: Palmas. 2009. vol 30, no. 1. p.55.
[3] MARIAU. Op. cit., p.26-28.
[4] HOYOS, Michel y RINCÓN, Víctor. Uso de dispositivos móviles para la captura de datos en campo con formularios electrónicos a través del programa Cybertracker. En: Palmas. 2014. vol 35, no. 4. p.127-136.

Para conseguir una observación continua de las actividades de plantación y expansión de los cultivos de palma de aceite que no requiera el desplazamiento del trabajador a través de toda la plantación, Pohl[5] describe un proyecto que busca desarrollar un sistema de monitoreo capaz de cuantificar las emisiones de dióxido de carbono extrayendo la información de un SAR (*Synthetic Aperture Radar*, Radar de Apertura Sintética) que opere en un satélite espacial. Sin embargo, este tipo de tecnologías son costosas y de difícil acceso y manejo para la mayoría de cultivadores en nuestra región.

En 2013, Patrick Ng *et al.*[6] afirmaron que mediante la adopción de tecnologías se puede conseguir una mayor productividad de los trabajadores y de las palmas, para lo cual plantean el uso de vehículos aéreos no tripulados (*UAVs, Unmanned Aerial Vehicles*), comparando algunos modelos existentes y resaltando su utilidad en la elaboración de mapas topográficos con información acerca de las condiciones del suelo.

Los estudios relacionados con la aplicación de UAVs en la industria agrícola han ido en aumento con el paso de los años. Fausto Costa *et al.*[7] utilizan estos vehículos para el esparcimiento de los químicos en los cultivos y evalúan su algoritmo de control para ajustar la ruta del UAV basándose en la información proveniente de sensores que, según lo demuestran, pueden minimizar la pérdida de pesticida. Por su parte Berni *et al.*[8] demuestran la utilidad de generar monitoreo remoto de productos usando un UAV para el control de la vegetación.

Al equipar las aeronaves con cámaras, surge una nueva perspectiva de la plantación y la posibilidad de utilizarla para el monitoreo de la vegetación ya que, tal y como lo afirman Patrick Ng *et al.*[9], las imágenes que se obtienen por este medio contrarrestan los problemas de alta nubosidad asociados con las imágenes satelitales, así como los periodos de retraso.

---

[5] POHL, Christine. Mapping palm oil expansion using SAR to study the impact on the CO_2 cycle. En: IGRSM International Remote Sensing & GIS Conference and Exhibition (7: 21-22, abril, 2014: Kuala Lumpur). IOP Conference Series: Earth and Environmental Science. 2014. vol. 20, no. 1. p. 1-8.

[6] NG, Patrick, *et al*. La teleobservación y las tecnologías digitales para el manejo de las plantaciones. En: Palmas. 2013. vol. 34, no. Especial, Tomo I. p. 261.

[7] COSTA, F.G, *et al.* The use of unmanned aerial vehicles and wireless sensor network in agricultural applications. En: Geoscience and Remote Sensing Symposium (IGARSS) (22-27, julio, 2012: Munich). IEEE International, 2012. p.5045-5048.

[8] BERNI, J, *et al.* Thermal and Narrowband Multispectral Remote Sensing for Vegetation Monitoring from an Unmanned Aerial Vehicle. En: IEEE Transactions on Geoscience and Remote Sensing. Marzo, 2009. vol.47, no.3, p.722-738.

[9] NG, Patrick, *et al*. Op. cit., p.270.

De esta forma, Gómez y Segura[10] plantean el uso de un UAV equipado con una cámara y la unidad de procesamiento de gráficos de un teléfono inteligente para resaltar, en las imágenes adquiridas por la aeronave, aquellas plantas que presentan un color amarillo en el borde de sus hojas, el cual es uno de los síntomas característicos de la pudrición del cogollo; esta situación es considerada, de acuerdo a la Federación Nacional de Cultivadores de Palma de Aceite (Fedepalma)[11], el principal problema sanitario de la palma de aceite en Colombia. Gómez y Segura[12] prueban el sistema en dos plantas domésticas diferentes y resaltan la importancia de implementar este tipo de sistemas directamente en plantaciones de palma de aceite y observar su comportamiento.

En el área de procesamiento de imágenes han sido numerosos los avances aplicados en la industria agrícola desde hace varios años. Nobou[13], propone el reconocimiento de la deficiencia de nutrientes en hojas utilizando el análisis del histograma de color RGB, mientras que Tian y Li[14] presentan el uso del "método de momento de color" para reconocer enfermedades en pepinos.

Sin embargo, desde inicios del siglo XXI, la detección de objetos en imágenes se ha visto enriquecida con el uso de algoritmos de aprendizaje automático que han permitido generar sistemas más robustos y generalizados. La principal aplicación en la agricultura se ha centrado en la evaluación del estado de maduración de los frutos producidos, como Zhang y McCarthy[15] que basados en imágenes por resonancia magnética y un modelo de clasificación centrado en un algoritmo de regresión de mínimos cuadrados parciales, logran clasificar tomates en diversos niveles de maduración.

Otras combinaciones de pre-procesamiento de la imagen con algoritmos de aprendizaje automático han sido aplicadas para el mismo tipo de evaluación en otros frutos: Guerrero y Benavides[16] se enfocan en el aguacate, Paulraj *et al.*[17] en

[10] GOMEZ, Sergio y SEGURA, Fredy. An aerial monitoring system (AMS) for detecting bud-rot diseases on oil palms. Tesis de Maestría. Bogotá D.C.: Universidad de los Andes.
[11] FEDEPALMA. Informe de Gestión 2012. [online]. Disponible en Fedepalma < http://issuu.com/fedepalma/docs/informe_de_gestio__n_fedepalma_2012>. p.61.
[12] GOMEZ, Sergio y SEGURA, Fredy. Op. cit., p.4-5.
[13] NOBOU, Spike. Application of information extraction in the plant growth based on image processing. En: Kansai Branch of the Agricultural Society of Machinery Engineers. 1992. vol. 72, p.63.
[14] TIAN, Y.W y LI, C.H. Research on Recognition of Cucumber Disease Based on Image Processing in Sunlight Greenhouse. En: Journal of Agricultural Mechanization Research. 2006. vol.2. p.151-153.
[15] ZHANG, Lu y MCCARTHY, Michael. Measurement and evaluation of tomato maturity using magnetic resonance imaging. En: Postharvest Biology and Technology. Mayo, 2012. vol.67, p.37-43.
[16] GUERRERO, Edgar y BENAVIDES, Gustavo. Automated system for classifying Hass avocados based on image processing techniques. En: IEEE Colombian Conference on Communications and Computing (COLCOM) (4-6, junio, 2014: Bogotá D.C., Colombia). Memorias. Bogotá D.C.: IEEE, 2014. p.1-6.
[17] PAULRAJ, M, *et al.* Color recognition algorithm using a neural network model in determining the ripeness of a banana. En: International Conference on Man-Machine Systems (ICoMMS) (11-13, octubre, 2009: Batu Ferringhi, Penang, Malasia). Proceedings. p.2B7-1 – 2B7-4.

las bananas, Rizam *et al.*[18] en la sandía y, Dadwal y Banga[19] en las manzanas. En el caso de las palmas de aceite, estas técnicas han sido aplicadas por Fadilah *et al.*[20], Bensaeed *et al.*[21], Jaffar *et al.*[22] y, May y Amaran[23] con diversas propuestas para la clasificación del fruto de la palma de acuerdo a su nivel de maduración.

Las técnicas de aprendizaje automático también han sido empleadas en imágenes aéreas con diferentes propósitos. Mnih y Hinton[24] se enfocan en la detección de calles en imágenes aéreas de alta resolución usando redes neuronales, mientras que Teutsch[25] procesa las imágenes provenientes del video de un UAV para la vigilancia de amplias áreas, ofreciendo la posibilidad de detectar y rastrear objetos móviles en la superficie terrestre mediante el uso de diversos enfoques como ventana deslizante y algoritmos NMS (*non-maximum suppression)*.

Esta combinación de procesamiento de imágenes y aprendizaje automático en imágenes aéreas ha sido también aplicada específicamente a las plantas. Aspinall [26] describe el uso de regresión logística para procesar imágenes hiperespectrales y detectar árboles caídos y diferentes especies de plantas en una zona de ribera en el Parque Nacional de Yellowstone en Wyoming, USA. Mientras tanto, Yang *et al.*[27] proponen un enfoque de detección automática de árboles en imágenes aéreas basados en un clasificador a nivel de pixel.

[18] RIZAM, Shah *et al.* Non-destructive watermelon ripeness determination using image processing and artificial neural network (ANN). En: International Journal of Electrical and Computer Engineering. vol.4. no.6.
[19] DADWAL, M. y BANGA, V. Estimate ripeness level of fruits using RGB color space and fuzzy logic technique. En: International Journal of Engineering and Advanced Technology. vol.2, no.1, p.225-229.
[20] FADILAH, Norasyikin, *et al.* Intelligent color vision system for ripeness classification of oil palm fresh fruit bunch. En: Sensors. 2012. vol.12, no.10, p.14179-14195.
[21] BENSAEED, O, *et al.* Oil palm fruit gradient using a hyperspectral device and machine learning algorithm. En: IOP Conference Series: Earth and Environmental Science. 2014. vol.20, no.1.
[22] JAFFAR, A, *et al.* Photogrammetric grading of oil palm fresh fruit bunches. En: International Journal of Mechanical and Mechatronics Engineering. Diciembre, 2009. vol.9, no.10, p.18-24.
[23] MAY, Z. y AMARAN, M. Automated ripeness assessment of oil palm fruit using RGN and fuzzy logic technique. En: WSEAS International Conference on Mathematical and Computational Methods in Science and Engineering (MACMESE) (13: 2011, Stevens Point, Wisconsin, USA). World Scientific and Engineering Academy and Society (WSEAS), 2011. p.52-59.
[24] MNIH, Volodymyr y HINTON, Geoffrey. Learning to detect roads in high-resolution aerial images. En: European Conference on Computer Vision (11: 5-11, septiembre, 2010: Heraklion, Crete, Greece). Proceeding. Berlin: Springer-Verlag, 2010. p.210-223.
[25] TEUTSCH, Michael. Moving object detection and segmentation for remote aerial video surveillance. Disertación para optar por el grado de Doctor de Ingeniería. Karlsruher Institut für Technologie (KIT). Facultad de Informática, 2014. p.242.
[26] ASPINALL, Richard. Use of logistic regression for validation of maps of the spatial distribution of vegetation species derived from high spatial resolution hyperspectral remotely sensed data. En: Ecological Modelling. 2002. vol.157, no.2-3, p.301-312.
[27] YANG, Lin, *et al.* Tree detection from aerial imagery. En: ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (17: 4-6, noviembre, 2009: Seattle, WA, USA). Proceedings. New York: ACM New York, 2009. p.131-137.

La aplicación de estas técnicas en la industria de la palma de aceite ha sido descrita por Kwesi[28] quien utiliza un modelo basado en un clasificador SVM (*Support Vector* Machine, Máquina de Vectores de Soporte) para mapear la distribución espacial de palmas de aceite en una región de Ghana usando imágenes provenientes del Landsat ETM+ (*Enhanced Thematic Mapper Plus*), el cual es un sensor a bordo del satélite Landsat 7 que obtiene imágenes de 8 bandas espectrales. A partir de este estudio concluye que una de las principales limitaciones de las imágenes satelitales fue la poca disponibilidad de información libre de nubes en las zonas de las plantaciones.

Finalmente, al revisar la literatura disponible, no se encuentran registros de trabajos de grados del nivel de pregrado relacionados con la aplicación específica de este proyecto.

[28] KWESI, Nooni. Oil palm mapping using Support Vector Machine with LANDSAT ETM+ data. Tesis de Maestría. Master of Science in Geo- Information Science and Earth Observation. Enschede, The Netherlands: Faculty of Geo-Information Science and Earth Observation of the University of Twente and the Faculty of Renewable Natural Resources of the Kwame Nkrumah University of Science & Technology, 2012. 68p.

3. JUSTIFICACIÓN

Uno de los principales problemas de la industria palmera ha sido siempre realizar labores de mantenimiento oportunas y eficientes que permitan obtener un alto nivel de productividad. Es por esto que se han realizado numerosos esfuerzos para implementar nuevas tecnologías que faciliten la obtención de la información disponible en las plantaciones y garanticen un mejor rendimiento, permitiendo que los cultivadores puedan administrar sus recursos de forma eficiente con el fin de lograr un desarrollo sostenible.

Una fuente significativa de información se obtiene a través del estudio de las fotografías aéreas tomadas por satélites espaciales, las cuales pueden ser obtenidas por medio de plataformas como Google Earth, sin embargo, estas imágenes no siempre brindan una solución óptima al problema, ya que obtener fotografías sin presencia de nubes en países tropicales como Colombia dificulta el proceso de adquisición de imágenes actualizadas, esto sin mencionar que dicho proceso puede tardar meses dependiendo de diversos factores como las condiciones del clima y la disponibilidad de satélites [29]. Para afrontar esta problemática, los vehículos aéreos no tripulados pueden ser considerados como una solución muy viable.

Estos vehículos controlados de forma remota generalmente son artefactos pequeños y sencillos de operar, son capaces de brindar registros fotográficos precisos ya que poseen la habilidad de obtener imágenes de alta resolución sobre terrenos de estudio más específicos, además, sus bajos costos de venta y mantenimiento los hacen fácilmente accesibles a todo tipo de público. Estas aeronaves representan una gran ventaja con respecto a los satélites ya que no solo son económicamente más rentables sino que también se encuentran disponibles en todo momento para ser utilizados en las labores de campo cada vez que se requiera.

Otro aspecto importante para la ejecución de posibles soluciones ante el problema que presentan las plantaciones de palma de aceite es el uso de un sistema de posicionamiento global (GPS) incorporado en el UAV. Este sistema permite realizar una distribución espacial precisa del terreno y le ofrece a la aeronave la posibilidad de realizar vuelos de forma totalmente autónoma guiado por coordenadas GPS, a través de rutas de vuelo previamente planeadas que abarcan

[29] NG., Patrick, *et al*. Op. cit., p.260.

grandes distancias y que representan una mejora significativa en el sistema de control del UAV, ya que permiten realizar vuelos con mayor suavidad y precisión. Todo esto con el objetivo de realizar mejores tomas que permitan obtener imágenes más estables y de mejor calidad para realizar un adecuado procesamiento de las mismas.

4. OBJETIVOS

4.1 OBJETIVO GENERAL

Desarrollar e implementar un software que permita procesar imágenes de una plantación de palma de aceite y registrar la cantidad de palmas cultivadas mediante el uso de un vehículo aéreo no tripulado y técnicas de visión artificial.

4.2 OBJETIVOS ESPECÍFICOS

- Investigar las tecnologías de UAVs disponibles actualmente en el mercado y definir el modelo más adecuado para las necesidades del proyecto.

- Desarrollar e implementar un software para procesar las imágenes que permita identificar y contar las palmas.

- Realizar pruebas de validación del sistema en plantaciones reales.

5. MARCO TEÓRICO

5.1 CULTIVO DE LA PALMA DE ACEITE

La palma de aceite o *Elaeis guineensis,* es originaria de las costas del Golfo de Guinea en el África occidental y fue introducida a América por los colonizadores y esclavos portugueses en el siglo XVI, llegando primero a Brasil. Su aparición en Colombia ocurrió en 1932 y fue sembrada inicialmente con fines ornamentales.

La industria palmera genera como productos del mercado: el aceite de palma o aceite de palmiste que se obtiene de las almendras de la palma y la torta de palmiste, un producto granular fino que resulta de este proceso de extracción. El 35% del mercado de aceite vegetal del mundo proviene de la palma de aceite ya que siendo cosechada durante todo el año, las palmas de aceite producen un promedio de 10 toneladas de fruto por hectárea, mucho más de lo que se obtiene con la soya, la colza o el girasol[30].

Este tipo de siembra pertenece a un cultivo tropical que crece idealmente a 28 grados Celsius con una altura sobre el nivel del mar inferior a los 500m, y que requiere de una disponibilidad de luz solar y de agua distribuida uniformemente en el tiempo para asegurar su maduración.

Colombia se considera en una posición geográfica privilegiada para la producción de palma de aceite, lo que la ha llevado a convertirse en el primer productor de aceite de palma en América y el cuarto a nivel mundial, después de Indonesia, Malasia y Tailandia.[31]

Para finales de 2014, el país tenía cerca de 500.000 hectáreas sembradas en palma de aceite, lo que significa que para 2016 debería poder contar con más de 2 millones de toneladas de aceite de palma duplicando la producción actual.[32]

---

[30]RSPO (Roundtable on Sustainable Palm Oil). Why is palm oil important? En: Green Palm Sustainability. 2014. Disponible en: <http://greenpalm.org/about-palm-oil/why-is-palm-oil-important>. Revisado en: Agosto, 2015.

[31] RSPO (Roundtable on Sustainable Palm Oil). Where is palm oil grown? En: Green Palm Sustainability. 2014. Disponible en: <http://greenpalm.org/about-palm-oil/where-is-palm-oil-grown-2>. Revisado en: Agosto, 2015.

[32] Portafolio. Colombia, cuarto productor de aceite de palma en el el mundo. 18, septiembre, 2014. Disponible en: <http://www.portafolio.co/especiales/portafolio-21-aniversario/colombia-productor-aceite-palma-2014> Revisado en: Agosto, 2015.

5.2 VEHÍCULOS AÉREOS NO TRIPULADOS

Inicialmente fueron denominados ROA, por sus siglas en inglés *"Remotely Piloted Aircraft"* (Aeronave Piloteada Remotamente). Sin embargo en la actualidad se utilizan los términos UAV (*"Unmanned Aerial Vehicle",* Vehículo Aéreo No Tripulado) o UAS ("*Unmanned Aircraft System"*, Sistema de Vuelo No Tripulado).

En forma general los UAVs se refieren a los vehículos aéreos sin tripulación humana a bordo, sin embargo este concepto no define el nivel de autonomía del mismo. Es por esto que Barrientos *et al*[33] definen un UAV o un UAS como una aeronave capaz de realizar una misión sin necesidad de tener una tripulación embarcada, sin excluir la existencia de un piloto que puede encontrarse en tierra. Mientras que definen un AAS (*Autonomous Aerial System,* Sistema Aéreo Autónomo o Aeronave Autónoma) como aquella que es capaz de desarrollar una misión sin necesidad de intervención humana.

Es posible clasificar a los UAVs de acuerdo al tipo de aeronave que representan, como se muestra en la Figura 1. Cada tipo de aeronave ofrece diferentes ventajas y características que la hacen ideal para distintas aplicaciones.

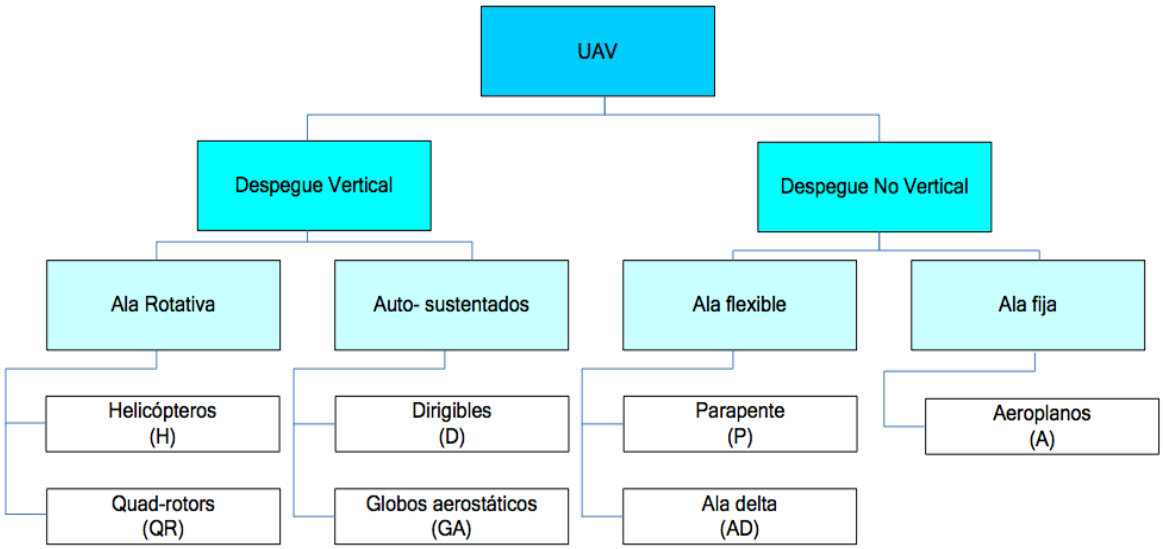
Las aplicaciones militares son las más conocidas y han sido el origen del desarrollo de esta tecnología. Sin embargo, con el paso de los años, se les ha dado uso en aplicaciones civiles, entre las que destacan: la inspección de infraestructuras y obras en construcción, la vigilancia de fronteras, el cine, la toma de datos en desastres naturales, la climatología, la agricultura, la búsqueda y rescate en naufragios o zonas de difícil acceso.

5.3 APRENDIZAJE AUTOMÁTICO

El aprendizaje automático (o *Machine Learning*, por su nombre en inglés) es una rama de la inteligencia artificial que desarrolla técnicas que le permiten a las computadoras aprender a solucionar problemas generalizando conocimiento a partir de situaciones dadas, de forma que puedan adaptarse a otras condiciones sin necesidad de ser reprogramadas.[34]

[33] BARRIENTOS, A, *et al*. Vehículos aéreos no tripulados para uso civil. Tecnología y Aplicaciones. Grupo de Robótica y Cibernética. Universidad Politécnica de Madrid. P. 2-9.
[34] BÉJAR, Javier. Aprendizaje automático. En: Diapositivas de curso (2014/2015: Barcelona, España). Inteligencia Artificial. Facultad de Sistemas Informáticos. Universitat Politècnica de Catalunya. p.4-6.

Figura 1. Clasificación de UAVs de acuerdo al tipo de aeronave



BARRIENTOS, A, *et al*. Vehículos aéreos no tripulados para uso civil. Tecnología y Aplicaciones. Grupo de Robótica y Cibernética. Universidad Politécnica de Madrid. p.3.

El aprendizaje se basa en un conjunto de datos llamados "conjunto de entrenamiento" (*training set*) a partir del cual se ajustan los parámetros de un modelo adaptativo. Esta fase es denominada "entrenamiento" o "fase de aprendizaje" y tiene como resultado una función $y(x)$ capaz de tomar un nuevo dato como entrada, proveniente de un conjunto desconocido para el programa llamado "conjunto de prueba o evaluación" (*test set*), y generar un vector de salida de acuerdo al tipo de predicción que se busca realizar. La habilidad de categorizar o predecir información correctamente con respecto a nuevas muestras diferentes a las que fueron usadas para entrenamiento, se le conoce como capacidad de generalización.[35]

Para la mayoría de aplicaciones, las variables de entrada originales son pre-procesadas para transformarlas a un nuevo espacio de variables, donde se espera que el problema de aprendizaje sea más fácil de resolver, e incluso puede acelerar el proceso de computación. Este pre-procesamiento es denominado "extracción de características" y aplica tanto para los datos de entrenamiento, como para los datos de prueba, e incluye, para el caso de imágenes, técnicas tales como procesamiento de imágenes, cambios de espacios de color[36], filtros o descriptores

[35] BISHOP, Christopher. Pattern Recognition and Machine Learning. Singapore: Springer, 2006. ISBN-10: 0-387-31073-8. p.2

[36] ANWER, Rao; VÁSQUEZ, David y LÓPEZ, Antonio. Color contribution to part-based person detection in different types of scenarios. En: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2011. vol.6855 LNCS, no.2, p.463-470.

como HOG (*Histogram of Oriented Gradients,* Histograma de Gradientes Orientados) o LBP (*Local Binary Pattern,* Patrón Binario Local).

Existen diversos tipos de enfoque para el aprendizaje automático[37]:

- **Aprendizaje supervisado:** Aplicaciones en las cuales el conjunto de entrenamiento incluye los vectores con las características de los datos de entrada junto a los correspondientes vectores de salida objetivos. Es decir, incluye el valor o clasificación "correcta" para cada dato de entrenamiento. El objetivo del aprendizaje puede ser:

  - Clasificación: asignar cada vector de entrada o muestra a un número finito de categorías discretas. Por ejemplo, el reconocimiento de dígitos o la clasificación de correos electrónicos no deseados.
  - Regresión: asignar cada vector de entrada a un valor de una o más variables continuas. Por ejemplo, la predicción del valor de una casa basado en sus características físicas y en el histórico del mercado.

- **Aprendizaje no supervisado:** Problemas en los cuales el conjunto de entrenamiento contiene los vectores de entrada sin los valores objetivo, es decir que no se cuenta con datos que definan si la información de salida es satisfactoria o no. El objetivo del aprendizaje puede ser:

  - Agrupamiento o "clustering": encontrar grupos de ejemplos similares en los datos de entrenamiento y agruparlos en distintas clases.
  - Estimación de densidad: determinar la distribución de la información en el espacio de las variables de entrada.
  - Visualización: proyectar la información de un espacio con múltiples dimensiones a uno de dos o tres dimensiones que permita su visualización.

- **Aprendizaje reforzado:** Problemas en los cuales se buscan acciones que ejecutar en situaciones dadas para maximizar una recompensa. El algoritmo no recibe ejemplos de salidas óptimas, como en el aprendizaje supervisado, sino que las descubre mediante un proceso de ensayo y error. Usualmente existe una secuencia de estados y acciones en las cuales el algoritmo interactúa con su ambiente y las acciones que toma tienen un

[37] BISHOP. Op. cit., p.3.

impacto en las recompensas durante cada paso. Por ejemplo, Tesauro[38] utiliza técnicas apropiadas de aprendizaje reforzado para que una red neuronal pueda aprender a jugar *backgammon* con un alto nivel de desempeño.

5.3.1 Scikit-learn[39]: Librería de aprendizaje automático en Python

Scikit-learn es un proyecto de software *open-source* que busca hacer que el aprendizaje automático sea accesible para todos, ya sea en la industria o en la academia, facilitando su aplicación en situaciones reales. Para esto provee una librería en el lenguaje de programación de propósito general Python, que ha sido ampliamente adoptado en el mundo científico y que es soportado por un próspero ecosistema de contribuidores.[40]

Este proyecto fue iniciado en 2007 como un proyecto desarrollado por David Cournapeau en el *Google Summer of Code*. Más tarde, ese mismo año, Matthieu Brucher empezó a trabajar en este proyecto como parte de su tesis. En 2010, Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort y Vincent Michel de INRIA (*Institut National de Recherche en Informatique et en Automatique,* Instituto Nacional de Investigación en Informática y Automática) tomaron el liderazgo del proyecto e hicieron su primer lanzamiento público el 1 de Febrero de 2010. Desde entonces, varias actualizaciones han ido apareciendo y la comunidad internacional ha liderado el desarrollo.[41]

Los algoritmos de aprendizaje en scikit-learn son incorporados como "estimadores", objetos instanciados con parámetros que controlan el aprendizaje. El conjunto de entrenamiento o *training data* es pasado a un método de ajuste llamado *fit method,* que acepta una matriz X de datos con la forma $(n \times p)$ donde n corresponde al número de muestras y p al número de características. Cuando existe una etiqueta de cantidad o clase correspondiente a la predicción correcta (como es el caso del aprendizaje supervisado), debe pasarse también un segundo argumento y, como un vector unidimensional con la forma $(n \times 1)$ que contiene la salida deseada para la correspondiente fila de X.

---

[38] TESAURO, Gerald. TD-Gammon, a self-teaching backgammon program, achieves master-level play. En: Neural Computation. 1994.

[39] PEDREGOSA, Fabian, *et al.* Scikit-learn: Machine Learning in Python. En: Journal of Machine Learning Research. 2011. vol.12, p.2825-2830.

[40] VAROQUAUX, Gaël, *et al.* Scikit-learn: Machine learning without learning the machinery. En: GetMobile: Computing and Communications. 2015. vol.19, no.1, p. 29.

[41] Scikit-learn, Sitio web. Historia, Documentación. Disponible en: <http://scikit-learn.org/stable/about.html>. Revisado en Agosto, 2015.

El rendimiento de un modelo mide su habilidad para predecir correctamente nueva información. Por tanto, debido al sobreajuste u *overfitting*, el conjunto de entrenamiento usado para aprendizaje generalmente no debe ser usado para evaluar el rendimiento del modelo, ya que puede resultar en una estimación excesivamente optimista del error de predicción del modelo. La forma correcta de obtener una estimación imparcial es dejar una porción de información llamada "conjunto de evaluación o prueba" (*test set*) que no es utilizada durante el entrenamiento y que es usada sólo para la evaluación del modelo.[42]

5.3.2 Regresión Logística

Modelo lineal de clasificación perteneciente al enfoque de aprendizaje supervisado. Se considera útil cuando la variable de salida o dependiente es categórica (por ejemplo, ser o no ser, presencia o ausencia) mientras que las variables explicativas o independientes son categóricas, numéricas o ambas.[43]

Como en el caso de la regresión lineal, se plantea una combinación lineal que considera las variables de influencias así:

$$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n, \quad (1)$$

en donde x_i (con $i = 1 \sim n$) es la variable independiente, es decir cada una de las características que definen una muestra; θ_i (con $i = 0 \sim n$) es el coeficiente de regresión para cada dato de muestra, los cuales corresponden a los parámetros que el modelo debe hallar durante el entrenamiento y a partir de los cuales generará las predicciones de las nuevas muestras; n corresponde al número de variables independientes, es decir, el número de características con las cuales se identifica una muestra; y finalmente z corresponde a la predicción de salida o variable dependiente, la cual es conocida durante el entrenamiento como *target*, *label* o etiqueta y será la respuesta que debe dar el modelo para el conjunto de prueba.

Sin embargo, el uso de una estructura matemática como la planteada en la ecuación 1, generará un modelo de regresión que puede arrojar cualquier tipo de valor continuo como respuesta. Dado que se busca, que el sistema tenga una variación de salida controlada, de forma que pueda establecerse un valor o umbral por encima del cual pertenece a una clase y por debajo del cual pertenece a otra,

[42] VAROQUAUX. Op. cit., p.31.
[43] MENARD, Scott. Applied logistic regression analysis. 2 ed. Thousand Oaks, CA: Sage University Papers Series on Quantitative Applications in the Social Sciences, 2002. p.7-106.

este modelo utiliza la forma de la función logística o sigmoide de la cual toma su nombre, convirtiéndose en:

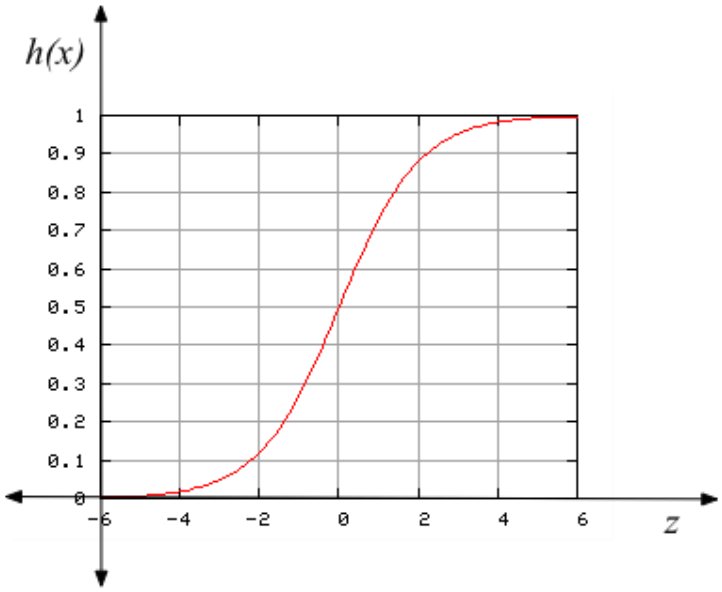
$$h_{\theta\,(x)} = \frac{1}{1+e^{-z}}, \tag{2}$$

en donde $h_{\theta(x)}$ representa la probabilidad que hay para la muestra x dada, de pertenecer a una clase representada por un valor de 1. De esta forma, la función z puede definirse como el logaritmo de esta razón, también llamado "logit", así:

$$z = \ln\left(\frac{h_{\theta\,(x)}}{1-h_{\theta\,(x)}}\right), \tag{3}$$

La Figura 2 presenta el comportamiento de la función sigmoide que modela la probabilidad $h_{\theta\,(x)}$ de pertenecer a una clase, con respecto a la función z que relaciona las características de la muestra con los parámetros del modelo.

Figura 2. Función sigmoide que modela la probabilidad $h_{\theta\,(x)}$ con respecto a z



Por tanto, si se considera que se usa el algoritmo para clasificar una muestra como perteneciente a una clase definida o no, se puede decir que: la condición $z = 0$ corresponde a la condición en la que la probabilidad de pertenecer a la clase es del 50%, ya que $h_{\theta\,(x)} = 0.5$. Si una muestra con un vector de características x obtiene un valor de $z > 0$ (es decir, $h_{\theta\,(x)} > 0.5$) tiene una probabilidad mayor del

50% de pertenecer a esta clase. Usualmente este es el umbral utilizado como punto de referencia para definir que una probabilidad por encima de este valor asegura que la muestra pertenece a la clase, y una por debajo de él lo excluye de la clase; sin embargo, el valor del umbral puede variar de acuerdo a la aplicación.[44]

5.4 PROCESAMIENTO DIGITAL DE IMÁGENES

Una imagen puede definirse como una función de dos dimensiones $f_{(x,y)}$, en donde x y y representan coordenadas en el plano espacial y el valor que toma la función determina la intensidad en ese punto, lo que más tarde corresponderá al color o nivel de gris. Cuando estos valores son limitados y discretos se considera que es una "imagen digital" en donde a cada punto se le denomina "pixel".

No existe un acuerdo general acerca de la diferencia entre procesamiento de imágenes y visión computarizada; sin embargo, Rafael González y Richard Woods [45] concluyen que el procesamiento de imágenes se define como "la disciplina en donde tanto la entrada como la salida del proceso son imágenes", mientras que la visión computarizada es un campo cuya "meta final es usar las computadoras para emular la visión humana, incluyendo el aprendizaje y la capacidad de hacer inferencias y tomar acciones basado en entradas visuales". Esta área es en sí misma, un campo de la Inteligencia Artificial que aún se encuentra en sus primeros pasos de desarrollo.

De acuerdo a Gary Bradsky y Adrian Kaehler[46], y en concordancia con la anterior diferenciación, la visión computarizada es la transformación de información que proviene de un video o de una cámara y se convierte en una decisión, como la cantidad de objetos presentes, o en una nueva representación como la conversión de una imagen en color a una imagen en grises.

Los procesos de modificación y análisis que se le aplican a una imagen dependen del requerimiento específico por el cual se desea trabajar con esa imagen; de forma general en la Figura 3 se mencionan algunas de las técnicas más comunes.

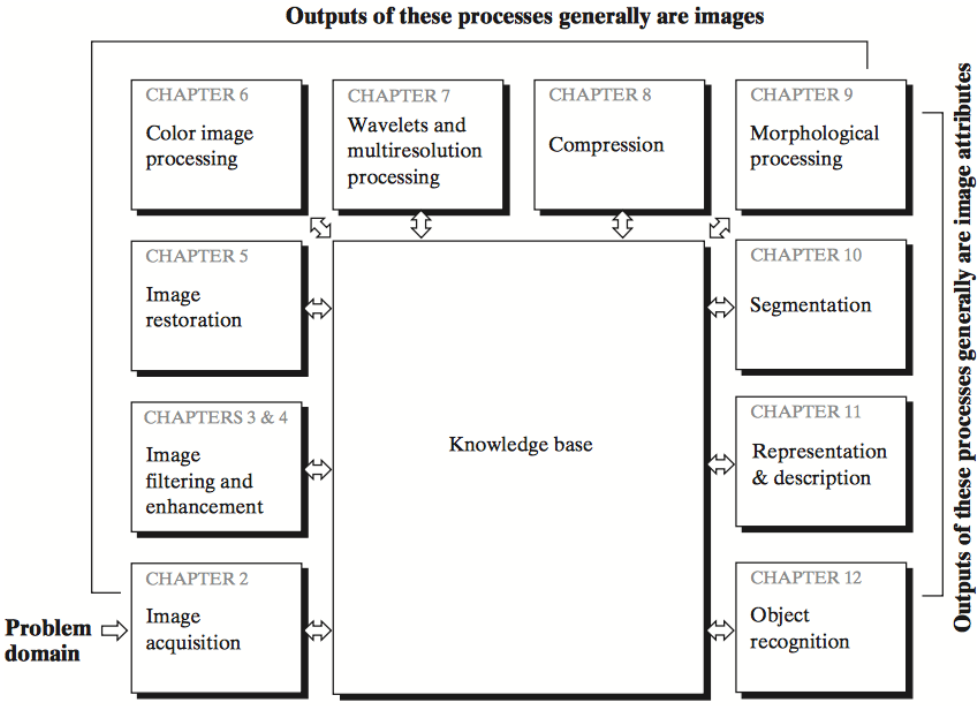
---

[44] DONG, Jia-Jyun, *et al.* Logistic regression model for predicting the failure probability of a landslide dam. En: Engineering Geology. Octubre, 2011. vol.117, no.1-2, p.53.

[45] GONZÁLEZ, Rafael C. y WOODS, Richard E. Digital Image Processing. 3 ed. New Jersey: Prentice Hall, Inc., 2008. p.2.

[46] BRADSKI, Gary y KAEHLER, Adrian. Learning OpenCV. Computer Vision with the OpenCV Library. Sebastopol: O'Reilly Media, Inc., 2008. p. 2.

Figura 3. Técnicas comunes de procesamiento de imágenes



GONZÁLEZ, Rafael C. y WOODS, Richard E. Digital Image Processing. 3 ed. New Jersey: Prentice Hall, Inc., 2008. p.2.

5.4.1 Scikit-image[47]: Librería de procesamiento de imágenes en Python

Scikit-image es una librería que implementa una colección de algoritmos de procesamiento de imagen en el lenguaje de programación Python y es desarrollada por una activa comunidad internacional de voluntarios y colaboradores. Está disponible bajo la licencia BSD Open Source.

La popularidad creciente de Python como un lenguaje de programación científico, junto a la progresiva disponibilidad de un largo ecosistema de herramientas complementarias, hacen que sea el ambiente ideal para manejar una herramienta de procesamiento de imagen.

El proyecto busca proveer algoritmos de alta calidad, bien documentados y con implementaciones sencillas de usar que faciliten la educación en procesamiento de imágenes, y por eso ha llegado a ser parte del programa anual *Google Summer of Code* donde los estudiantes participantes aprenden acerca de procesamiento

[47] WALT, Stéfan van der, *et al.* Scikit-image: Image processing in Python. En: PeerJ 2:e453. Abril, 2014. Disponible en: <http://dx.doi.org/10.7717/peerj.453> Revisado en: Agosto, 2015.

de imagen e ingeniería de software a través de sus contribuciones al proyecto. Así mismo, busca proveer a la industria con una forma confiable de atacar problemas sin tener que gastar una cantidad significante de energía en la reimplementación de algoritmos que ya se encuentran disponibles en paquetes comerciales.

El proyecto scikit-image comenzó en agosto de 2009 y ha recibido contribuciones de más de 100 personas. El paquete puede ser instalado en la mayoría de las plataformas (como BSD, GNU/Linux, OS X, Windows).

Entre las funciones de los sub-módulos contenidos en el paquete se encuentran: conversión de espacio de color, manejo de archivos de imágenes, dibujos primitivos como líneas y textos, ajustes de intensidad, manejo de histogramas, filtros, visualización de figuras, operaciones morfológicas, restauración, segmentación, transformaciones geométricas, detección y extracción de características, análisis de texturas, entre otros.

Las imágenes son representadas como NumPy arrays (el estándar *de facto* para almacenar información multidimensional en Python científico). Cada array tiene: un número de dimensiones específico, 2 para el caso de una imagen 2D en escala de grises, 3 para una imagen 2D multicanal, o 4 para una imagen 3D multicanal; una forma (M, N, dimensión) para una imagen con M pixeles verticales y N pixeles horizontales; y un tipo de dato numérico como `float` para pixeles de valores continuos o `unit8` para pixeles de 8 bits.

El uso de NumPy arrays como la estructura de información fundamental maximiza la compatibilidad con el resto del ecosistema científico de Python, de forma que los datos pueden ser pasados inmediatamente a otras herramientas como NumPy, SciPy, matplotlib, scikit-learn, OpenCV, y otros.

5.4.2 *Local Binary Pattern o LBP* (Patrón Binario Local)

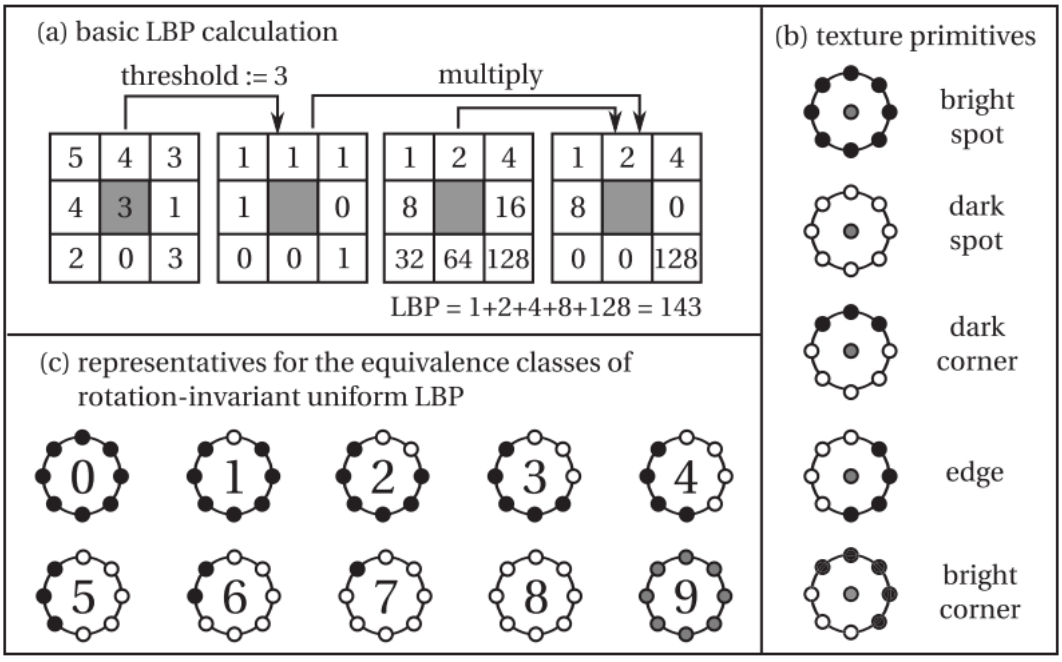
Es un operador que transforma una imagen en un array o imagen de etiquetas enteras que describen la apariencia de la misma[48]. Ha tenido una amplia aplicación en el área de investigación en procesamiento de imágenes para clasificación de texturas, detección de caras o modelamiento de fondo. Es fácil de implementar, rápido de computar y describe las características de una imagen de una forma robusta y con alto rendimiento.[49] Adicionalmente, entre sus principales

[48] PIETIKÄINEN, Matti, *et al.* Computer vision using local binary patterns. Springer, 2011. ISBN:9780857297471.
[49] TEUTSCH. Op. cit. p.67.

ventajas está su invariancia a la traslación de la imagen y a los cambios monotónicos del nivel de gris.[50]

La estructura clásica del cálculo del LBP se observa en la Figura 4(a). El valor de grises del pixel central es comparado con cada uno de sus vecinos: si el valor del vecino es mayor o igual, la posición de este vecino será indicada con 1, y si por el contrario, el vecino tiene un valor menor que el pixel central, tomará un valor de 0. La codificación LBP es calculada al multiplicar todas las posiciones indicadas de la vecindad con sus pesos relativos y sumándolos, de forma que cada vecino representa 1 bit, y al tener una vecindad de 8, pueden obtenerse 8 bits y conformar un byte como lo muestra la Figura 5. El valor obtenido será el que reemplace al pixel central en la nueva matriz e informe de la relación específica de ese pixel con sus vecinos, como se observa en la Figura 6. Este valor será un número entero en el rango de [0 - 255].[51]

Figura 4. Cálculo e interpretación del LBP

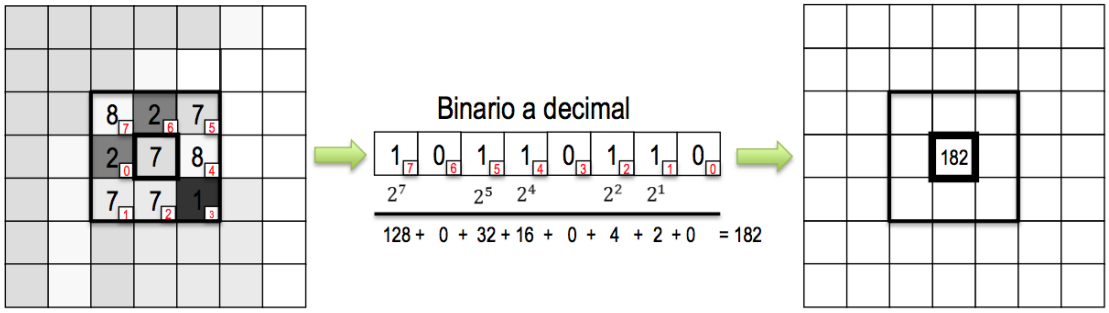


TEUTSCH, Michael. Moving object detection and segmentation for remote aerial video surveillance. Disertación para optar por el grado de Doctor de Ingeniería. Karlsruher Institut für Technologie (KIT). Facultad de Informática, 2014. p.68.

[50] LÓPEZ, Antonio; VALVENY, Ernest y VANRELL, María. CLASIFICACIÓN DE OBJETOS. Local Binary Patterns. [Diapositivas] En: Detección de objetos [Curso online]. Universitat Autònoma de Barcelona: mayo-julio, 2015. Disponible en: Coursera <https://www.coursera.org/course/deteccionobjetos>

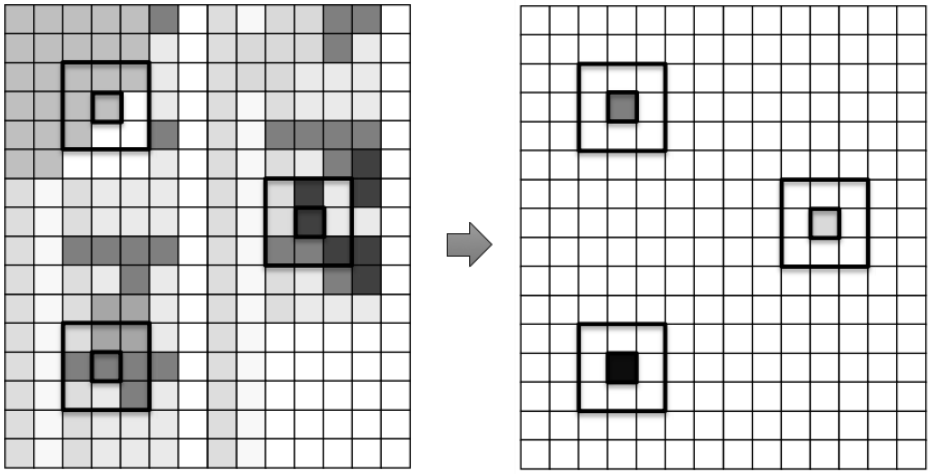
[51] TEUTSCH. Op. cit. p.67.

Figura 5. Pesos relativos de cada vecino y conformación del byte para el pixel central



LÓPEZ, Antonio; VALVENY, Ernest y VANRELL, María. CLASIFICACIÓN DE OBJETOS. Local Binary Patterns. [Diapositivas] En: Detección de objetos [Curso online]. Universitat Autònoma de Barcelona: mayo-julio, 2015. Disponible en: Coursera <https://www.coursera.org/course/deteccionobjetos>

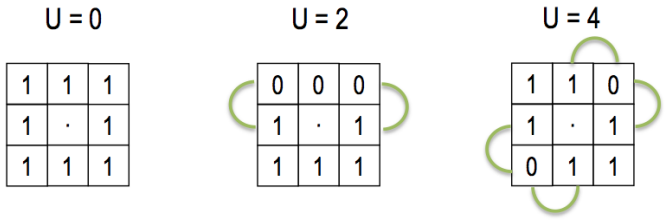
Figura 6. Valor LBP para cada pixel en la nueva imagen LBP



LÓPEZ, Antonio; VALVENY, Ernest y VANRELL, María. CLASIFICACIÓN DE OBJETOS. Local Binary Patterns. [Diapositivas] En: Detección de objetos [Curso online]. Universitat Autònoma de Barcelona: mayo-julio, 2015. Disponible en: Coursera <https://www.coursera.org/course/deteccionobjetos>

A partir de este concepto, existen diversas variantes. Una de las más importantes es el LBP Uniforme que busca reducir el número de patrones, ya que con la idea básica cada imagen analizada tiene 256 posibles patrones para cada pixel. Para lograr la disminución utiliza una medida de uniformidad U que representa el número de transiciones 1/0 o 0/1 que se dan en cada vecindad LBP como lo muestra la Figura 7.

Figura 7. Medida de uniformidad usada en LBP Uniforme



LÓPEZ, Antonio; VALVENY, Ernest y VANRELL, María. CLASIFICACIÓN DE OBJETOS. Local Binary Patterns.- Variantes (LBP Uniforme) [Diapositivas] En: Detección de objetos [Curso online]. Universitat Autònoma de Barcelona: mayo-julio, 2015. Disponible en: Coursera <https://www.coursera.org/course/deteccionobjetos>

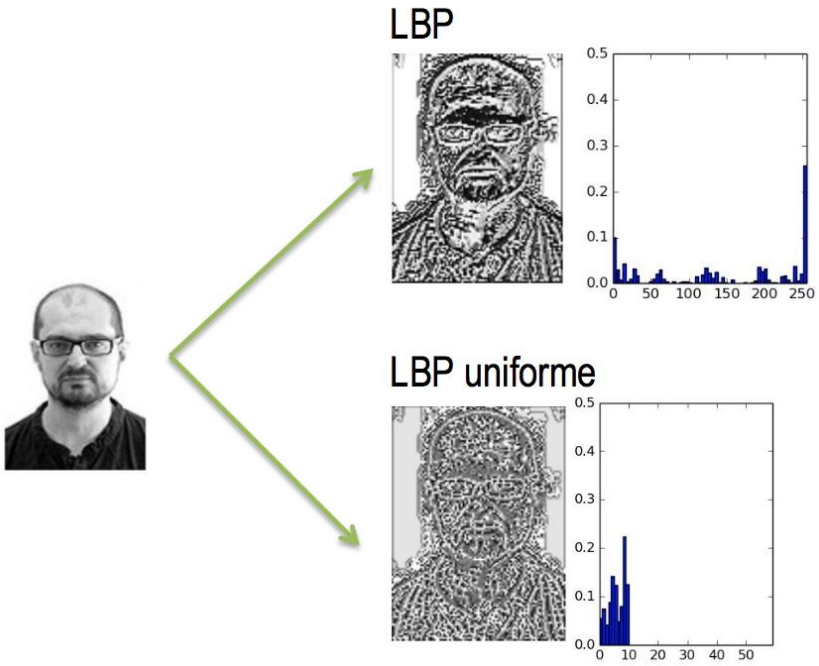
A los patrones con U=0 o U=2 se les asigna un código de patrón individual como los observados en la Figura 4(c) ya que estos representan las texturas primitivas como las observadas en la Figura 4(b), que se consideran esenciales para la interpretación de la textura. Al resto de patrones se les asigna el mismo código y pasan a ser indistinguibles.

Como se observa en la Figura 4(c) existen nueve clases equivalentes (clases 0 a 8) y 8 posibles codificaciones LBP en cada clase: una LBP por cada rotación en pasos de 45º. Se da una excepción en las clases *bright spot* (punto brillante) y *dark spot* (punto oscuro), ya que sus clases respectivas (0 y 8) sólo tienen un posible código LBP debido a que la rotación no representa ningún cambio. La décima clase (9) contiene todos los otros 198 LBP que no son texturas primitivas.[52] De esta forma se obtienen un total de 59 posibles patrones que serán los valores que puede tomar cada pixel en la nueva imagen LBP-U, en vez de los 256 posibles en la forma tradicional, generando una disminución significativa de datos como se observa en la Figura 8.

Adicionalmente, el LBP-U tiene la cualidad de ser invariante a la rotación ya que le asigna toda las posibles rotaciones a la misma clase, por ejemplo, las texturas *edge, bright corner* y *dark corner* (borde, esquina brillante y esquina oscura) son representadas, respectivamente, por las clases equivalentes 4, 3 y 5 sin importar la rotación que tengan.

[52] Íbid., p.67.

Figura 8. Comparación de histogramas LBP y LBP-U



LÓPEZ, Antonio; VALVENY, Ernest y VANRELL, María. CLASIFICACIÓN DE OBJETOS. Local Binary Patterns – Variantes (LBP Uniforme). [Diapositivas] En: Detección de objetos [Curso online]. Universitat Autònoma de Barcelona: mayo-julio, 2015. Disponible en: Coursera <https://www.coursera.org/course/deteccionobjetos>

6. ESTRUCTURA GENERAL DEL SISTEMA

El sistema de conteo planteado se basa en 3 bloques fundamentales, cuya estructura general se presenta en la Figura 9.

Figura 9. Diagrama de bloques de la estructura general del sistema

Para la captura de la información se utiliza un vehículo aéreo no tripulado de referencia DJI Phantom 2 Vision+ controlado mediante la aplicación Pix4Dmapper Capture (desarrollada por la empresa Pix4D) instalada en un dispositivo móvil con sistema operativo iOS, en la cual se establece el área de la plantación que será recorrida por el drone, la ruta definida y los puntos en los cuales adquirirá fotografías. Las imágenes son almacenadas para su posterior procesamiento en una memoria microSD Lexar de escritura rápida y alto rendimiento.

Una vez que todas las zonas de interés han sido recorridas por el UAV y la información proveniente de ellas ha sido almacenada, las imágenes y los datos de geolocalización son descargados en la computadora y procesados mediante el software de fotogrametría Pix4Dmapper con el objetivo de crear ortomosaicos de las zonas de interés, permitiendo que cada palma pueda ser observada desde el aire bajo la misma proyección ortogonal y sin efectos de perspectiva, lo que facilita el conteo de las mismas. De esta forma, se obtiene finalmente una imagen por cada lote o zona de interés.

Posteriormente, todas las imágenes de interés son enviadas al software desarrollado en este proyecto, el cual las evalúa y genera como resultado un reporte con el número de palmas encontradas y la imagen de cada zona con los cuadros de detección correspondientes que delimitan la ubicación de las palmas localizadas.

7. SELECCIÓN DEL UAV

Para realizar el proceso de adquisición de imágenes, el cual consiste en obtener fotografías aéreas de la plantación junto con sus datos de geolocalización, se hace estrictamente necesario definir qué modelo de UAV disponible en el mercado es el más adecuado para realizar este proceso. Para ello se hace un análisis comparativo entre algunos de los modelos más destacados, contando únicamente con aquellos que integran una cámara fotográfica o que tienen la posibilidad de adaptarla en su estructura, y teniendo en cuenta los aspectos más importantes según las necesidades del proyecto, tales como el costo, tipo de cámara, tiempo de vuelo, compatibilidad con herramientas de fotogrametría y la posibilidad de realizar vuelos de forma autónoma. También es importante mencionar que uno de los factores que más inciden en este proceso es la adaptabilidad del UAV al terreno o zona de estudio en la cual se desea implementar.

7.1 FACTORES QUE INCIDEN EN EL PROCESO DE SELECCIÓN

7.1.1 Tipo de despegue (vertical o no vertical)

Teniendo en cuenta que las plantaciones de palmas de aceite están distribuidas por lotes, los cuales en la mayoría de los casos solo son accesibles por medio de una carretera o camino de tierra que los subdivide, el tipo de despegue y aterrizaje del UAV se convierte en un factor decisivo, pues es necesario que la aeronave pueda adaptarse a esta situación, la cual es más crítica cuando las palmas de estos lotes son de edad avanzada, ya que esto implica que las ramificaciones de aquellas cercanas al camino sobresalen sobre este dejando aún más limitado el espacio. Debido a esta problemática los UAVs de despegue vertical, tales como los helicópteros y multirotores optan por ser la solución más adecuada.

7.1.2 Vuelo autónomo

Debido al limitado espacio que existe entre los lotes de palmas, y a la altura que pueden desarrollar, superando más de 12 metros, se hace prácticamente imposible controlar el UAV manualmente desde tierra, pues la línea de vista entre el piloto y la aeronave queda obstaculizada ante este escenario. Para resolver este problema es necesario que los vuelos se realicen de forma autónoma mediante rutas previamente programadas en el UAV. Esta funcionalidad representa una ventaja significativa para el proyecto, pues generalmente se implementa a través de un sistema GPS incorporado en el UAV. Esto facilita el

proceso de adquisición de imágenes, pues permite programar los puntos donde deben ser adquiridas las fotografías y al mismo tiempo georreferenciarlas, optimizando el proceso de elaboración de mapas u ortomosaicos a través de algunas herramientas de fotogrametría. Teniendo en cuenta esto, se hace imprescindible escoger un modelo que posea esta habilidad.

7.1.3 Cámara

La cámara es otro de los factores fundamentales que inclinan la balanza hacia algún modelo en particular. En el caso de las plantaciones de palmas de aceite, y por tratarse de tomas aéreas, el área de interés se encuentra justo bajo la ruta de vuelo del UAV. Por ello es importante escoger un modelo con una cámara, ya sea integrada o adaptable, que pueda orientarse apuntando directamente a tierra para formar una proyección ortogonal entre su línea de vista y el suelo. Adicional a esto, debe tenerse en cuenta la resolución en megapíxeles, ya que a la hora de elaborar los mapas de las zonas de interés, las herramientas de fotogrametría deberán ser capaces de extraer suficientes "*keypoints*" o puntos claves, basándose únicamente en la información disponible en cada pixel para realizar el emparejamiento de una sucesión de imágenes. Es decir, que a mayor resolución, mayor será el número de "*keypoints*" extraídos por imagen, lo que optimiza el proceso y genera mejores resultados al obtener mapas de alta calidad y sin problemas de distorsión.

7.1.4 Autonomía de vuelo

El tiempo que un UAV es capaz de mantenerse en vuelo con una única carga de batería es un factor que puede ayudar a reducir la carga de trabajo y minimizar costos. Tener la capacidad de mantener la aeronave en vuelo durante más tiempo genera la posibilidad de abarcar áreas más grandes y realizar el proceso de adquisición de datos de manera rápida y eficiente. Por lo general, la autonomía de vuelo de la mayoría de modelos de aeronaves multirotores disponibles en el mercado oscila entre los 12 y 15 minutos. Algunas empresas fabricantes de UAVs, tales como DJI, han logrado mejorar considerablemente este aspecto, ofreciendo algunos modelos que garantizan tiempos de hasta 25 minutos de vuelo. Es importante tener esto en cuenta para decidir cuál fabricante ofrece la mejor opción.

7.1.5 Estabilidad y maniobrabilidad

Debido a que el UAV debe volar en campo abierto, es importante que este pueda ofrecer ciertas habilidades de vuelo que le permitan desplazarse de manera eficiente. La capacidad de mantener su curso estable, ya sea estando en movimiento o en una posición fija (para el caso de los multirotores) es importante para garantizar que las fotografías sean tomadas con excelente precisión.

Además, hay que tener en cuenta que existen varios factores externos que afectan la maniobrabilidad, tales como el viento y la capacidad de carga útil que puede levantar el UAV, siendo este último el principal problema de aquellos modelos a los cuales se les adapta una cámara externa o accesorios adicionales que incrementan su peso comprometiendo la capacidad de respuesta de la aeronave.

7.1.6 Costo

Uno de los factores más importantes a analizar es el costo que pueda demandar un UAV, no solo por su precio de venta sino también por sus labores de mantenimiento. Existen muchos modelos extremadamente costosos y con infinidad de accesorios, instrumentos y cámaras muy avanzadas, que en realidad pueden ser totalmente innecesarios o de poca utilidad para el proyecto que se desea realizar. Lo ideal es desarrollar un sistema de bajo costo que represente una mejora significativa en el proceso que se desea automatizar para que se justifique el valor de su inversión.

7.1.7 Herramientas de fotogrametría

Los UAVs no necesariamente deben ser compatibles de manera directa con las herramientas o softwares de fotogrametría, pues estos necesitan básicamente fotografías y datos de geolocalización para crear los ortomapas que se desean obtener, por tanto, los UAVs son solo un medio para adquirir esta información. Sin embargo, plantear rutas de vuelo que sean adecuadas para que estos datos sean recolectados de manera correcta requiere de cierta habilidad y conocimiento. Es por esto que algunos desarrolladores de software y fabricantes de UAV crean cada vez más herramientas, softwares o aplicaciones que permitan programar estas aeronaves para seguir rutas de vuelo especialmente diseñadas con este fin.

7.2 COMPARACIÓN DE ALGUNOS MODELOS DISPONIBLES

7.2.1 Especificaciones técnicas

Basados en las características previamente mencionadas, se realiza un cuadro comparativo con las especificaciones técnicas de 3 referencias de UAVs con los que cuenta la Universidad Pontificia Bolivariana (Seccional Bucaramanga) para determinar si alguno de ellos es apto para cumplir a cabalidad con los objetivos del proyecto. Se escogieron únicamente UAVs de despegue vertical por ser los más adecuados para el tipo de entorno en el cual se desean implementar. La Figura 10 muestra el cuadro comparativo realizado para estos modelos.

Figura 10. Comparación de modelos de UAV disponibles en la Universidad Pontificia Bolivariana (Seccional Bucaramanga)

	 **AR.Drone**	 **AR.Drone 2.0**	 **Phantom 1 + GoPro HERO3+**
Cámara	Cámara frontal y vertical (integradas)	-Cámara frontal y vertical (integradas)	Cámara GoPro HERO3+- Black Edition (adaptable con soporte)
Video	- Cámara frontal: 480p-15fps - Cámara vertical: QCIF de 60fps para medir la velocidad de avance	- Cámara frontal: HD 720p-30 fps - Cámara vertical: QVGA de 60fps para medir la velocidad de avance	4Kp-15fps / 2.7Kp-30fps / 1440p-48fps / 1080p-60fps / 960p-100fps / 720p-120fps
Foto	NO	Cámara frontal: 1280x720 pixeles Cámara vertical: 640 x 480 pixeles	12 MP
Wi-Fi	SI	SI	- Phantom 1: NO - GoPro HERO3+: SI
GPS	NO	SI (por medio del accesorio "Flight Recorder")	SI
Vuelo Autónomo	NO	SI (por medio del accesorio "Flight Recorder")	SI (solo para volver al punto de despegue en caso de pérdida de señal o batería baja)
Medio de control	Dispositivo móvil con sistema operativo Android o iOS mediante conexión Wi-Fi	Dispositivo móvil con sistema operativo Android o iOS mediante conexión Wi-Fi	Control Remoto (2.4GHz)
Rango de Control	50m	50m	1000m
Velocidad Máxima	10m/s	10m/s	10m/s
Peso	- 380g con carcasa para exteriores - 420g con carcasa para interiores	- 380g con carcasa para exteriores - 420g con carcasa para interiores	- 800g sin cámara - 890g con cámara
Tiempo de Vuelo	12 minutos	12 minutos	-15 minutos (sin cámara) -12 minutos (con cámara)

7.2.2 Pruebas de vuelo

Se realizan pruebas de vuelo con cada uno de estos modelos con el fin de determinar cuál ofrece las mejores prestaciones. Se descarta inicialmente el modelo AR.Drone por ser el más limitado de las 3 referencias, ya que no posee la capacidad de tomar fotografías ni implementar un sistema GPS. La Figura 11 expone las ventajas y desventajas que se observaron en los modelos AR.Drone 2.0 y Phantom 1 luego de realizar varias prácticas de vuelo.

7.3 SELECCIÓN DEL MODELO

Luego de analizar en detalle la temática expuesta anteriormente, y de estudiar las ventajas y desventajas que ofrecen los modelos AR.Drone 2.0 y Phantom 1 + GoPro HERO3^{+}, se decide, que de los modelos disponibles en el mercado para uso civil, el UAV más adecuado para la ejecución del proyecto es el Phantom 2 Vision+ del fabricante DJI. Dado que es un modelo con el que no se contaba inicialmente, se realizaron los trámites necesarios para que la Universidad Pontificia Bolivariana (Seccional Bucaramanga) pudiese adquirirlo y así continuar con la siguiente fase del proyecto. En la Figura 12 se exponen las características que ofrece este UAV.

Figura 11. Ventajas y desventajas observadas durante las prácticas de vuelo para los modelos AR.Drone 2.0 y Phantom 1 + GoPro HERO3+

	 AR.Drone 2.0		 Phantom 1 + GoPro HERO3+	
	Ventajas	**Desventajas**	**Ventajas**	**Desventajas**
Despegue y Aterrizaje	Despegue y aterrizaje totalmente automáticos	Algunas veces el aterrizaje es muy fuerte	Despegue muy estable y con potencia	Requiere algo de experiencia y habilidad por parte del piloto, sobretodo en el procedimiento de aterrizaje
Vuelo Autónomo	- Mejora considerablemente la estabilidad en vuelo - Permite enviar el UAV a una posición específica por medio del control por mapa - Modo "Return Home" que permite enviar el UAV de vuelta al punto de despegue	- Requiere "Flight Recorder" (accesorio adicional) - Sólo es posible establecer un "Waypoint" (punto de ruta)	En caso de pérdida de señal o batería baja, el UAV vuelve automáticamente al punto de despegue	No es posible realizar vuelos autónomos con rutas programadas
Cámara	- Posee dos cámaras, una frontal y otra vertical - Permite la visualización de video en tiempo real con ambas cámaras - La cámara vertical tiene línea de vista directa con el suelo	La cámara vertical posee una resolución muy baja	- Phantom1: Permite adaptar diferentes tipos de cámaras - GoPro HERO3+: Alta resolución de fotos y videos - GoPro HERO3+: Permite la visualización de video en tiempo real mediante un dispositivo móvil con conexión Wi-Fi	No es posible posicionar la cámara en un ángulo de 90º con respecto al suelo
Telemetría	Visualización en tiempo real de la altura, posición, velocidad y carga de la batería			Internamente el UAV mide su posición, altura, velocidad y carga de batería pero no es posible visualizar estos datos por parte del piloto
Autonomía de Vuelo		- Poca autonomía de vuelo (máximo 12 minutos) - Al volar a altas velocidades la autonomía de vuelo disminuye considerablemente (promedio de 7 minutos)	- 15 minutos de vuelo (UAV sin cámara) - Con la cámara GoPro HERO3+ el tiempo de vuelo disminuye, pero se mantiene en un rango aceptable. (promedio de 12 minutos) - Los vuelos a altas velocidades no representan una pérdida significativa de energía (13 minutos sin cámara y 10 minutos con cámara)	
Estabilidad y Maniobrabilidad	Excelente estabilidad y maniobrabilidad en interiores	En exteriores la estabilidad del UAV se ve bastante limitada. Incluso los vientos leves a menudo son suficiente para desviarlo de su curso	- Excelente maniobrabilidad y tiempo de respuesta - La estabilidad del UAV es excelente en exteriores, siendo capaz de mantener su rumbo aún en presencia de vientos leves - El posicionamiento por medio de GPS le permite permanecer en una posición fija con bastante precisión	No es apto para volar en interiores con espacio reducido

Figura 12. Descripción del modelo de UAV seleccionado para el proyecto

	 **Phantom 2 Vision+**	
Cámara	- Cámara integrada, estabilizada con cardán electrónico de 3 ejes - Control de inclinación con rango de 0° - 90°	
Video	1080p - 30fps / 720p - 60fps	
Foto	14 MP	
Wi-Fi	SI	Incluye “Wi-Fi Range Extender” que permite un rango de conexión de 500m - 700m (campo abierto)
GPS	SI	
Vuelo Autónomo	SI	- Permite programar rutas de vuelo hasta con 16 “waypoints” (puntos de ruta) - En caso de pérdida de señal o batería baja, el UAV vuelve automáticamente al punto de despegue
Velocidad Máxima	15m/s	
Peso	1242g	
Tiempo de Vuelo	25 minutos	
Telemetría	Datos de posición, altura, velocidad y carga de la batería se pueden visualizar en tiempo real	
Medio de Control	- Control remoto (5.728 GHz – 5.85 GHz) - Dispositivos móviles Android y iOS a través de la Aplicación “DJI Visión” y múltiples aplicaciones de otros desarrolladores - PC Ground Station (accesorio adicional) - iPad Ground Station (accesorio adicional)	
Rango de Control	- Control remoto: 800m (campo abierto) - Dispositivos móviles Android y iOS a través del “Wi-Fi Range Extender”: 700m (campo abierto) - PC Ground Station: 15km (campo abierto) - iPad Ground Station: 2 km (campo abierto)	
Herramientas de Fotogrametría	SI	Compatibilidad con Pix4D y DroneDeploy

8. CREACIÓN DE ORTOMOSAICOS

Siendo el conteo de las palmas el objetivo fundamental del proyecto, se hace estrictamente necesario determinar qué tipo de imágenes son las más adecuadas para facilitar la labor del software de detección y conteo y así obtener los mejores resultados. Teniendo en cuenta esto, se decide que los ortomosaicos son el tipo de imagen más apropiada para este proceso debido a que permiten observar todos sus elementos bajo una misma proyección ortogonal y sin efectos de distorsión de perspectiva.

La Figura 13 muestra una fotografía tomada por el UAV en determinado sector de la plantación. En esta imagen se observa claramente el problema de distorsión de perspectiva que genera la cámara a bordo del UAV. Esto se debe al lente gran-angular con el que comúnmente vienen equipadas estas cámaras, dichos lentes ofrecen un ángulo de visión muy amplio pero a cambio de ello causan distorsiones en la imagen. Esto tiene sus ventajas y desventajas para ciertos campos de aplicación, pero para el caso particular de detectar y contar las palmas no resulta muy útil. Tal como se observa en la Figura 13, las palmas son fácilmente reconocibles solo en la zona central de la fotografía, pero a medida que nos acercamos a los bordes, la distorsión de perspectiva aumenta gradualmente hasta el punto en que se hace cada vez más difícil distinguir o diferenciar una palma de otra.

En el caso de un ortomosaico las fotografías son procesadas para generar una imagen como la que aparece en la Figura 14. Sin embargo para lograr esto, un ortomosaico debe ser creado a partir de una sucesión de imágenes con un porcentaje de solapamiento constante entre ellas. Para esto es necesario utilizar diferentes herramientas de fotogrametría que permitan obtener estas imágenes de manera adecuada y posteriormente procesarlas para la creación de los respectivos ortomosaicos.

Figura 13. Imagen adquirida por el UAV en determinada zona de la plantación

Figura 14. Ortomosaico generado para determinada zona de la plantación mediante el software de fotogrametría Pix4Dmapper

8.1 SOFTWARE DE FOTOGRAMETRÍA

Para la elaboración de los ortomosaicos se decide utilizar el software de fotogrametría “Pix4Dmapper” de la compañía Pix4D. Este software permite procesar una gran cantidad de imágenes y convertirlas, según las necesidades del usuario, en un modelo digital de superficie (DSM), una nube de puntos para la creación un modelo del terreno en 3D, o un ortomosaico.

La compañía Pix4D también ofrece la aplicación móvil “Pix4Dmapper Capture”, la cual es compatible con dispositivos de sistema operativo Android y iOS. Esta aplicación permite programar rutas de vuelo especialmente diseñadas para proyectos de fotogrametría en el UAV Phantom 2 Vision+, facilitando el proceso de adquisición de datos, los cuales son guardados directamente en un archivo de proyecto de Pix4Dmapper (.p4d) que puede ser ejecutado inmediatamente después de descargar los datos a la computadora.

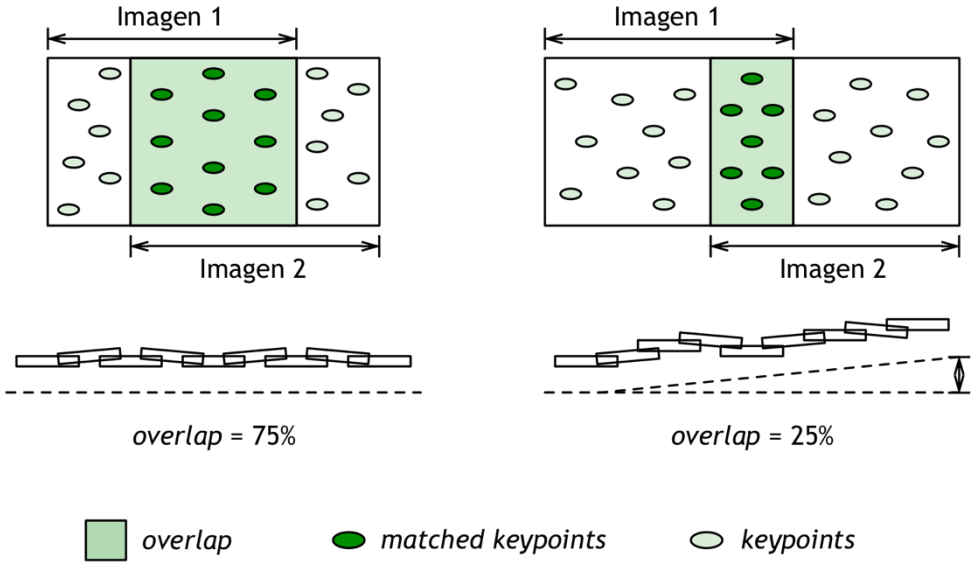
8.1.1 Funcionamiento del software Pix4Dmapper

Pix4Dmapper es un software de procesamiento de imágenes que se encarga de encontrar automáticamente miles de puntos en común existentes entre imágenes. Cada punto característico encontrado en una imagen se le denomina *keypoint (*punto clave*).* Cuando 2 *keypoints* en 2 imágenes diferentes son encontrados y se determina que son el mismo punto, son considerados como *matched keypoints (*puntos clave emparejados*)*. Cada grupo de *matched keypoints* que coinciden correctamente generará un punto 3D. Cuando existe un alto porcentaje de *overlap* (solapamiento) entre dos imágenes, el área común entre ellas es más grande y por tanto más *keypoints* pueden ser emparejados. Es decir, mientras mayor sea el número de *keypoints*, mayor será la precisión de los puntos 3D que pueden ser computados. Esta es la razón por la cual la regla principal es mantener un alto porcentaje de *overlap* entre las imágenes.[53]

En la Figura 15 se puede observar la relación directa que existe entre el *overlap* y los *matched keypoints*, y también se puede apreciar como surgen los errores de distorsión que se propagan a lo largo del proceso de emparejamiento de las imágenes como consecuencia directa de un bajo porcentaje de *overlap*.

[53] Pix4D. Selecting the Images Acquisition Plan Type. En: Pix4D Support Site. Agosto, 2015. Disponible en: <https://support.pix4d.com/hc/en-us/articles/202557459#label7>. Revisado en: Septiembre, 2015.

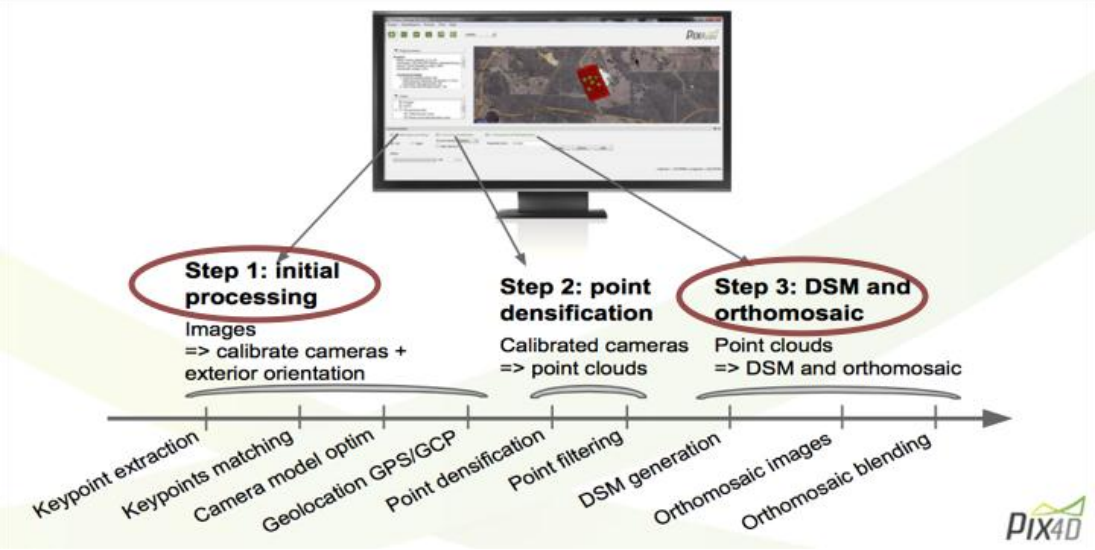
Figura 15. Relación entre el porcentaje de *overlap* y el número de *matched keypoints*



Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: <https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.12.

La Figura 16 muestra el diagrama general de proceso que implementa el software Pix4Dmapper. En ella se resaltan dos de las tres etapas de procesamiento, la de procesamiento inicial y la de generación de DSM y ortomosaico, que son los que realmente se necesitan para esta fase del proyecto.

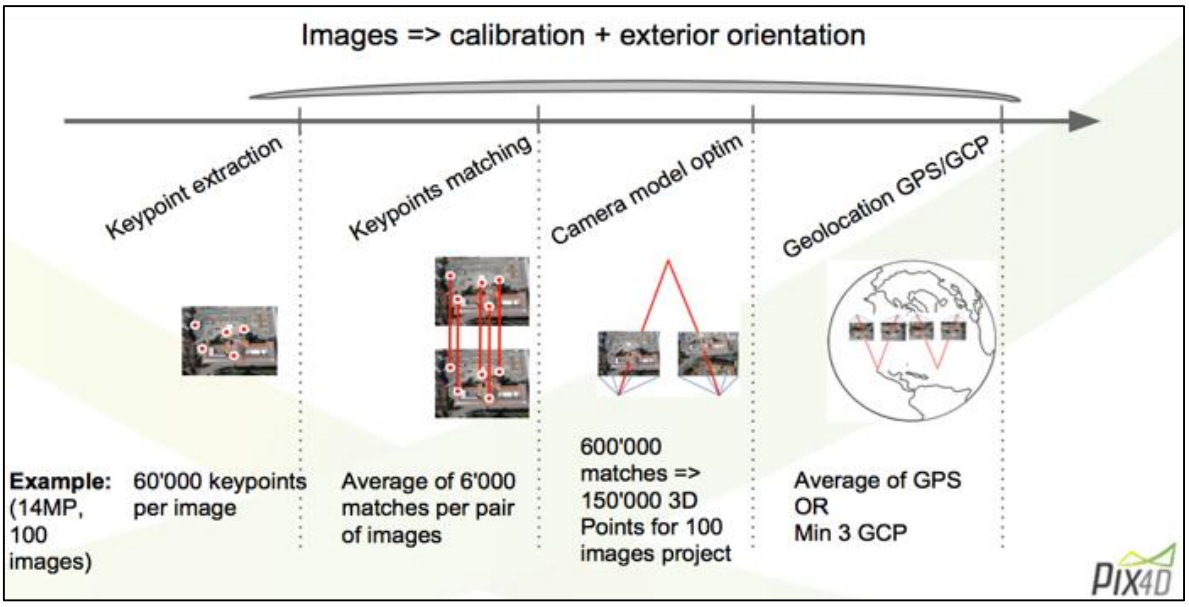
Figura 16. Diagrama general de proceso del software Pix4Dmapper



Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.9.

La primera etapa que aparece resaltada en la Figura 16, denominada procesamiento inicial, es la etapa más importante del proceso, pues de ella dependerán las otras 2. Es aquí en donde se realiza el proceso de extracción y emparejamiento de *keypoints*, explicado anteriormente, que permite la creación de la nube de puntos 3D, el modelo digital de superficie y el ortomosaico. La Figura 17 muestra paso a paso cómo se realiza la etapa del procesamiento inicial.

Figura 17. Diagrama general del procesamiento inicial que se realiza en el software Pix4Dmapper



Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.11.

8.2 ADQUISICIÓN DE IMÁGENES Y DATOS DE GEOLOCALIZACIÓN

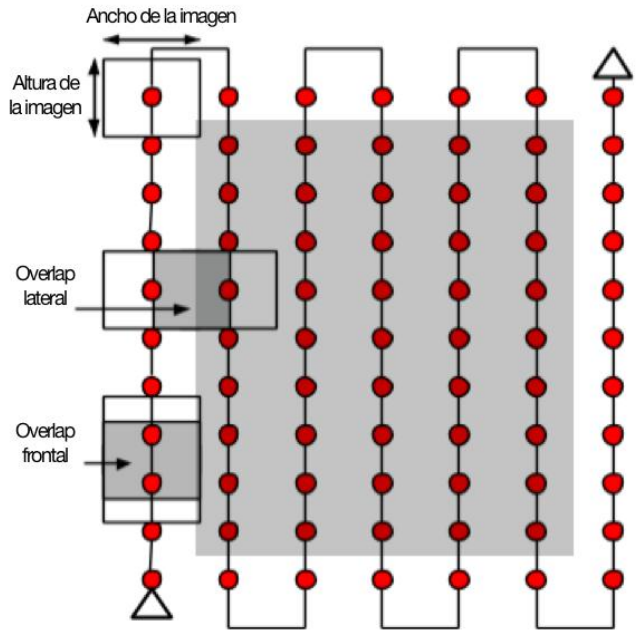
8.2.1 Diseño del plan de vuelo

La planificación del vuelo puede ser considerada una de las fases más importantes, pues las decisiones que aquí se tomen tendrán consecuencias en las demás etapas del proyecto. Si la ruta de vuelo programada en el UAV no es la adecuada para las condiciones del terreno, los ortomosaicos que se esperan obtener al procesar las imágenes con el software de fotogrametría, no tendrán la calidad necesaria para ser utilizados por el software de detección y conteo de palmas. Es importante tener en cuenta que en el proceso de planificación se deben contemplar los desniveles del terreno para garantizar un vuelo cuya altura sea constante en todas las áreas de interés. Esto se hace con el fin de mantener un nivel de resolución homogéneo en todas las zonas del ortomosaico resultante.

Para realizar un plan de vuelo exitoso debe tenerse en cuenta las siguientes recomendaciones, tal como se muestra en la Figura 18:

- Las imágenes deben ser tomadas siguiendo un patrón de cuadrícula regular.
- El UAV debe mantenerse a una altura constante a lo largo de toda la ruta.
- El *overlap* recomendado que deben tener las fotografías es de al menos un 75% de *overlap* frontal y un 60% de *overlap* lateral. Este valor es el más óptimo para la mayoría de los casos pero puede variar dependiendo de la complejidad del terreno.[54]

Figura 18. Plan de vuelo ideal para la mayoría de los casos



Pix4D. Selecting the Images Acquisition Plan Type. En: Pix4D Support Site. Agosto, 2015. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202557459#label7>. Revisado en: Septiembre, 2015.

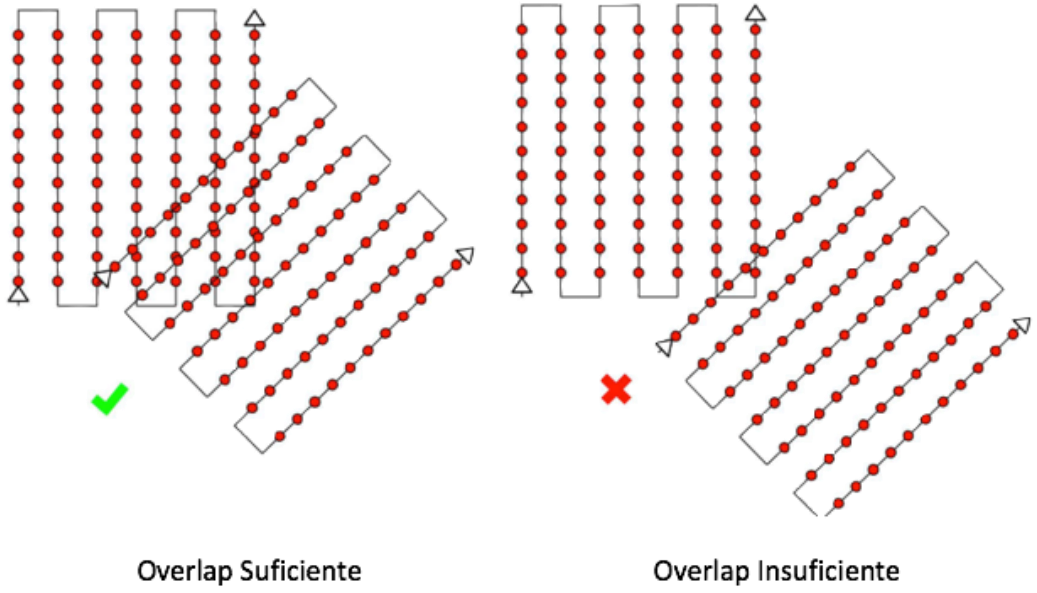
En caso de realizar múltiples vuelos deben tenerse en cuenta las siguientes recomendaciones:

- Cada plan de vuelo debe ser capaz de obtener imágenes con un alto porcentaje de *overlap*.
- Debe haber suficiente *overlap* entre cada plan de vuelo.
- Los distintos vuelos deben ser realizados a la misma altura.

[54] Pix4D. Selecting the Images Acquisition Plan Type. En: Pix4D Support Site. Agosto, 2015. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202557459#label7>. Revisado en: Septiembre, 2015.

La Figura 19 muestra cómo debe ser el nivel de *overlap* entre varias cuadrillas de vuelo, mientras que la Figura 20 expone cómo es el plan de adquisición de imágenes más recomendado para este fin.

Figura 19. *Overlap* entre distintos planes de vuelo



Pix4D. Selecting the Images Acquisition Plan Type. En: Pix4D Support Site. Agosto, 2015. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202557459#label7>. Revisado en: Septiembre, 2015.

Figura 20. Plan de adquisición de imágenes recomendado para dos vuelos diferentes

Pix4D. Selecting the Images Acquisition Plan Type. En: Pix4D Support Site. Agosto, 2015. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202557459#label7>. Revisado en: Septiembre, 2015

El plan de vuelo ideal para el proceso de adquisición de imágenes depende del tipo de terreno que se desea mapear. Por ello es importante saber qué tipo de imágenes son consideradas fáciles o difíciles de procesar por el software de fotogrametría a la hora de realizar la extracción de *keypoints.*

La Figura 21 muestra algunas imágenes que se consideran óptimas para ser procesadas, por ejemplo, aquellas con alto contenido visual, tales como:

- Imágenes con arbustos, rocas o suciedad
- Imágenes de edificios o zonas urbanas
- Imágenes con una resolución mayor a 10MP

Figura 21. Imágenes consideradas óptimas para el proceso de extracción de *keypoints*

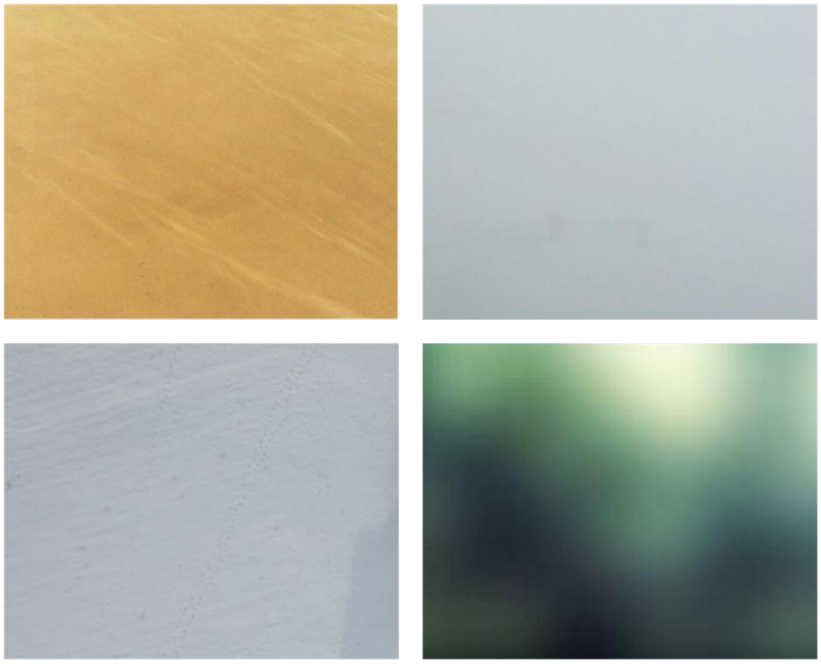
Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.14.

La Figura 22 muestra imágenes que son consideradas difíciles de procesar debido a que representan texturas complejas o no poseen una resolución adecuada, tales como:

- Imágenes con presencia de arena, niebla o nieve
- Imágenes borrosas o desenfocadas
- Imágenes sobreexpuestas a demasiada luz
- Imágenes tomadas con poca luminosidad
- Imágenes con una resolución menor a 3MP

Tener imágenes a las cuales se les puedan extraer suficientes *keypoints* no siempre es suficiente para obtener buenos resultados. Por eso también es importante saber qué tipo de imágenes son fáciles, difíciles o incluso imposibles de emparejar.

Figura 22. Imágenes consideradas difíciles para el proceso de extracción de *keypoints*



Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.14.

La Figura 23 representa aquellas imágenes que se pueden emparejar fácilmente, la Figura 24 aquellas que son difíciles y la Figura 25 las que le resultan imposibles para el software de fotogrametría.

Figura 23. Imágenes consideradas fáciles de emparejar tras el proceso de extracción de *keypoints*

Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.15.

Figura 24. Imágenes consideradas difíciles de emparejar tras el proceso de extracción de *keypoints*

Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.15.

Figura 25. Imágenes consideradas imposibles de emparejar tras el proceso de extracción de *keypoints*

Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide >. Revisado en: Agosto, 2015. p.15.

Teniendo en cuenta que las plantaciones de palma de aceite corresponden a zonas clasificadas como boscosas o de vegetación densa, se considera que las imágenes que deben ser adquiridas por el UAV representarán un desafío a la hora de ser procesadas por el software de fotogrametría. Esto se debe a que la textura compleja de los árboles y la densa vegetación frecuentemente ocasiona que estos se vean muy diferentes entre cada par de imágenes solapadas. Para lograr que

estas imágenes puedan arrojar los mejores resultados a la hora de crear los ortomosaicos, se establece que la ruta de vuelo óptima debe realizarse siguiendo un patrón de de cuadricula regular como se plantea en la Figura 18, pero teniendo en cuenta las siguientes consideraciones:

- El *overlap* debe establecerse en un mínimo de 85% de *overlap* frontal y un 70% de *overlap* lateral para lograr un mayor emparejamiento de *keypoints.*
- A medida que aumenta la altitud la distorsión de perspectiva disminuye y la densa vegetación representa mejoras en sus propiedades visuales, es decir, se hace más fácil detectar similitudes entre cada par de imágenes solapadas. Por ello se establece la altura de vuelo a un mínimo de 80m.[55]

8.2.2 Ejecución del plan de vuelo y recolección de datos

Una vez definidas las especificaciones del plan de vuelo, se utiliza la aplicación Pix4Dmapper Capture instalada en un dispositivo móvil con sistema operativo iOS para programar la ruta de vuelo que debe recorrer el Phantom 2 Vision+. Esto se realiza mediante una conexión Wi-Fi entre el dispositivo móvil y la aeronave, siendo estrictamente necesario que el vuelo se realice dentro de un rango de distancia que pueda garantizar una conexión confiable y sin interrupciones. Esto se debe a que la acción de tomar una foto no es controlada directamente por el Phantom 2 Vision+ sino que es enviada de manera automática por medio de la aplicación Pix4Dmapper Capture, la cual calcula el momento exacto en el que debe ser tomada una foto basándose en los datos de velocidad y posición que recibe el UAV, es decir, que si la conexión falla, la toma de fotos también se interrumpe, provocando pérdida de datos importantes para el proceso de creación de ortomosaicos. Sin embargo, el Phantom 2 Vision+ es capaz de volar de manera totalmente autónoma y seguir con la ruta previamente establecida aun si se interrumpe la conexión.

La Figura 26 muestra la interfaz de esta aplicación en la sección de "*Map View"* donde se ajusta el tamaño de la cuadrícula y se establece la altura de vuelo. Por otro lado la Figura 27 muestra la sección de configuraciones en donde se establecen parámetros como el *overlap* y la velocidad.

[55] Pix4D. Selecting the Images Acquisition Plan Type. En: Pix4D Support Site. Agosto, 2015. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202557459#label7>. Revisado en: Septiembre, 2015.

Figura 26. Interfaz de la aplicación Pix4Dmapper Capture en la sección de "*Map View*"

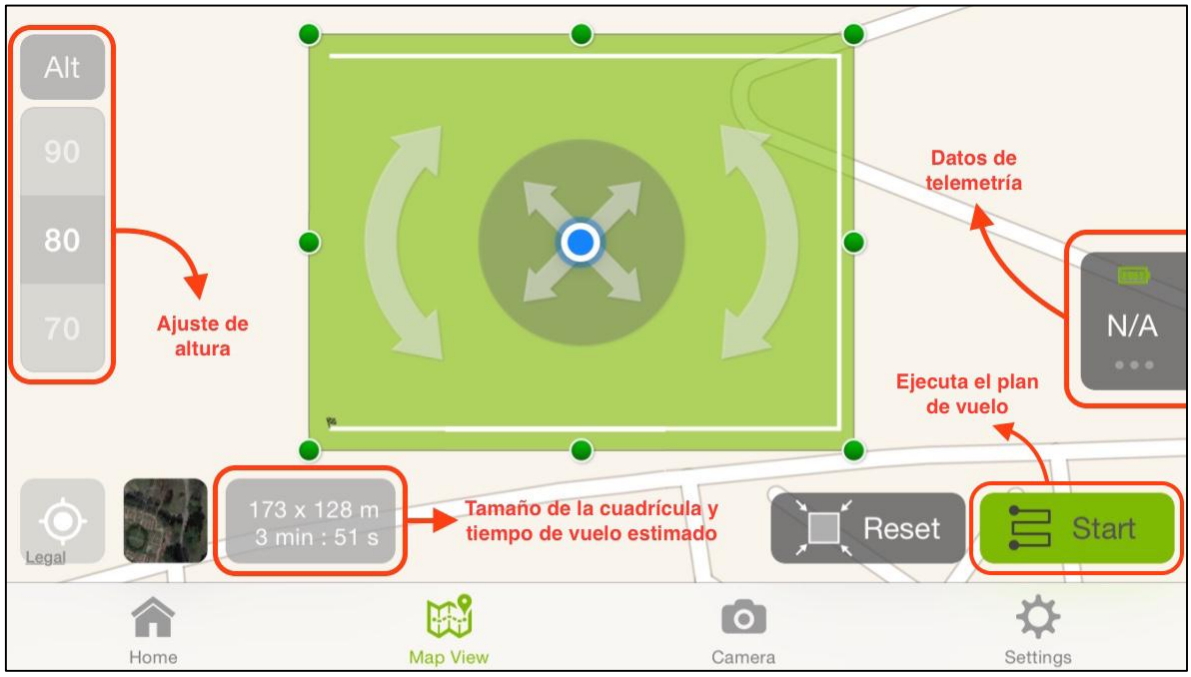
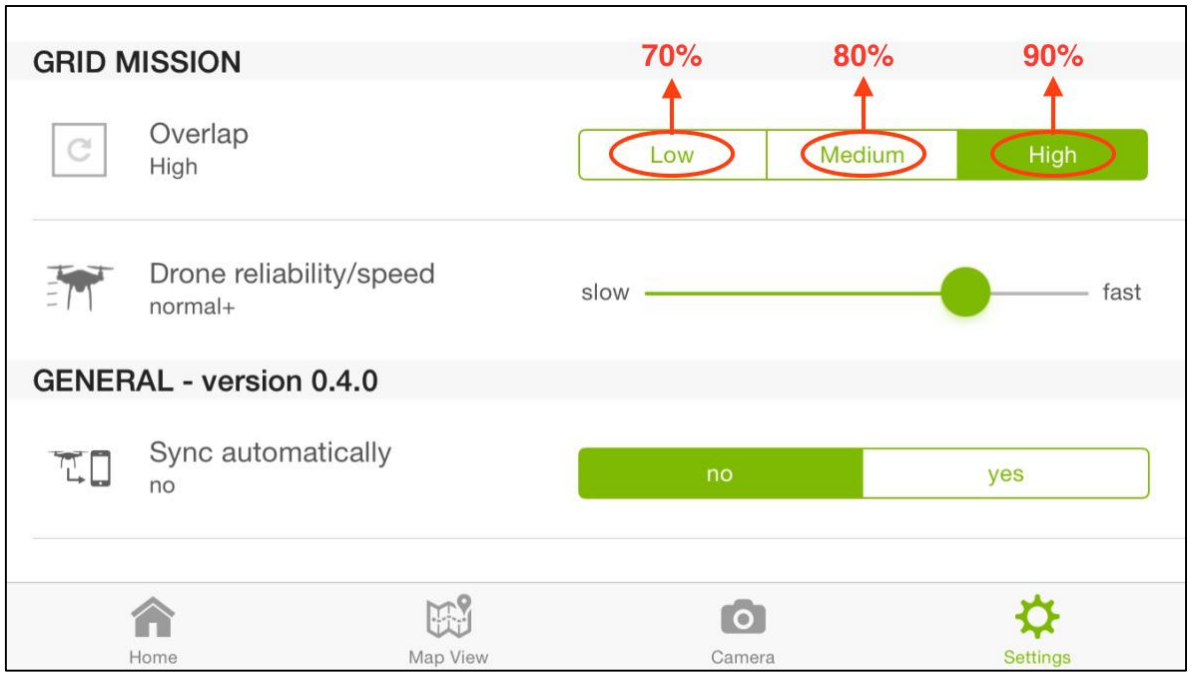
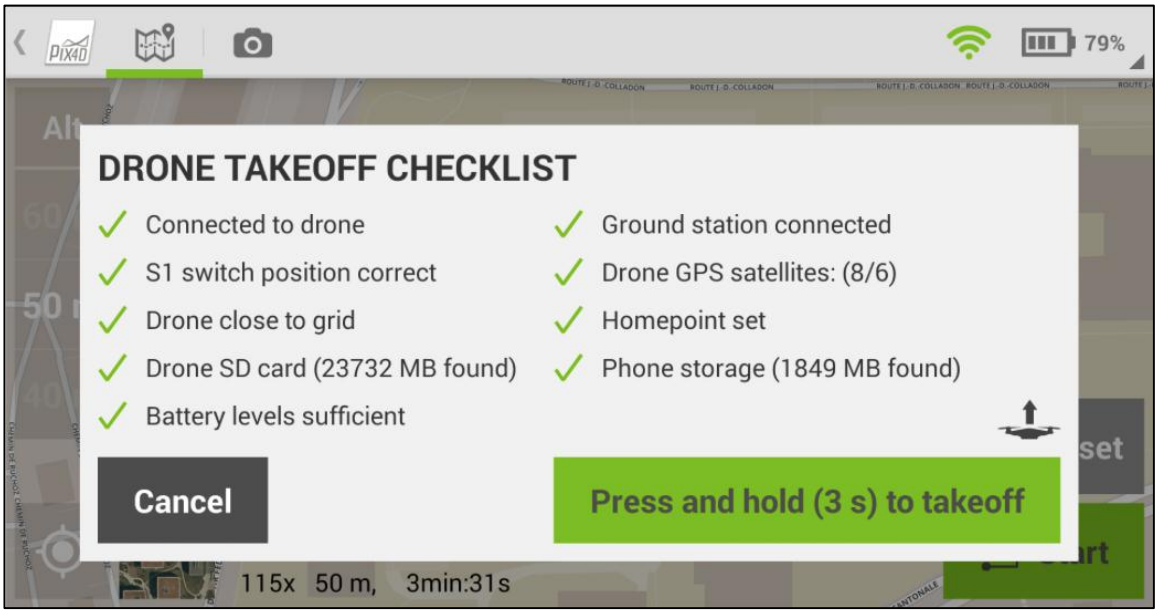


Figura 27. Interfaz de la aplicación Pix4Dmapper Capture en la sección de "*Settings*"



Luego de establecer los ajustes del plan de vuelo en la aplicación Pix4Dmapper Capture, se procede a la ejecución del vuelo oprimiendo el botón "*Start*" que aparece en la Figura 26. Antes de iniciar el vuelo, la aplicación comprueba que los parámetros que aparecen en la Figura 28 sean válidos para proceder con el despegue.

Figura 28. Lista de parámetros a validar antes de proceder con el despegue del UAV



Pix4D. (Android) Pix4Dmapper Capture App - Getting started. En: Pix4D Support Site. Mayo, 2015. Disponible en: < https://support.pix4d.com/hc/en-us/articles/202557269--Android-Pix4Dmapper-Capture-App-Getting-started#label6>. Revisado en: Agosto, 2015.

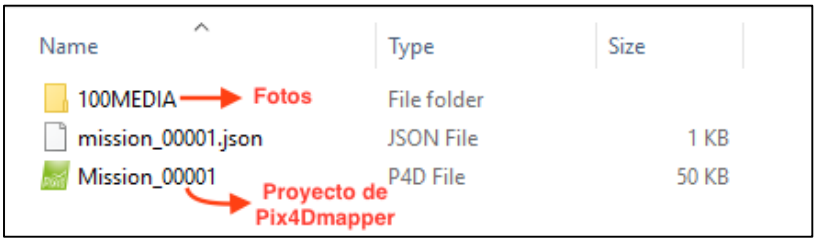
Una vez que esta lista es revisada, la aeronave levanta vuelo de manera automática y recorre la ruta programada recolectando las fotografías necesarias para la creación del ortomosaico. Esta fase se encuentra totalmente automatizada, por tanto solo hay que esperar que el UAV termine su recorrido para proceder con la descarga de los datos a la computadora. Durante este proceso la aplicación permite visualizar en tiempo real el movimiento de la aeronave a lo largo de su ruta tal como se observa en la Figura 29.

Una vez que el UAV realiza su recorrido, las imágenes y los archivos de proyecto de Pix4Dmapper deben ser descargados a una computadora para su posterior procesamiento. Las fotografías quedan almacenadas en la memoria microSD del Phantom 2 Vision+, mientras que los archivos de proyecto de Pix4Dmapper se almacenan en la memoria del dispositivo móvil. La Figura 30 muestra cómo lucen estos datos al ser descargados en la computadora.

Figura 29. Visualización en tiempo real de la ruta de vuelo del UAV

Pix4D. (iOS) Pix4Dmapper Capture App - Getting started. En: Pix4D Support Site. Agosto, 2015. Disponible en: < https://support.pix4d.com/hc/en-us/articles/204692015--iOS-Pix4Dmapper-Capture-App-Getting-started>. Revisado en: Agosto, 2015.

Figura 30. Fotografías y archivos de Pix4Dmapper descargados a la computadora



8.3 PROCESAMIENTO DE LOS DATOS EN EL SOFTWARE PIX4DMAPPER

8.3.1 Procesamiento inicial

Una vez recolectados los datos se utiliza el software Pix4Dmapper para procesar las imágenes y los datos de geolocalización. La Figura 31 muestra cómo luce la interfaz del software cuando se ejecuta un archivo de proyecto de Pix4Dmapper.

En la Figura 32 se muestra cómo el software administra los datos recolectados por el UAV. Allí se observa claramente cómo cada fotografía se encuentra relacionada con sus respectivos datos de geolocalización, indicando para cada una de ellas su latitud, longitud, altitud, precisión horizontal y vertical y los ángulos omega, phi y

kappa que corresponden a la inclinación del UAV en cada uno de sus ejes X, Y y Z al momento de tomar cada imagen.

Figura 31. Interfaz del software Pix4Dmapper

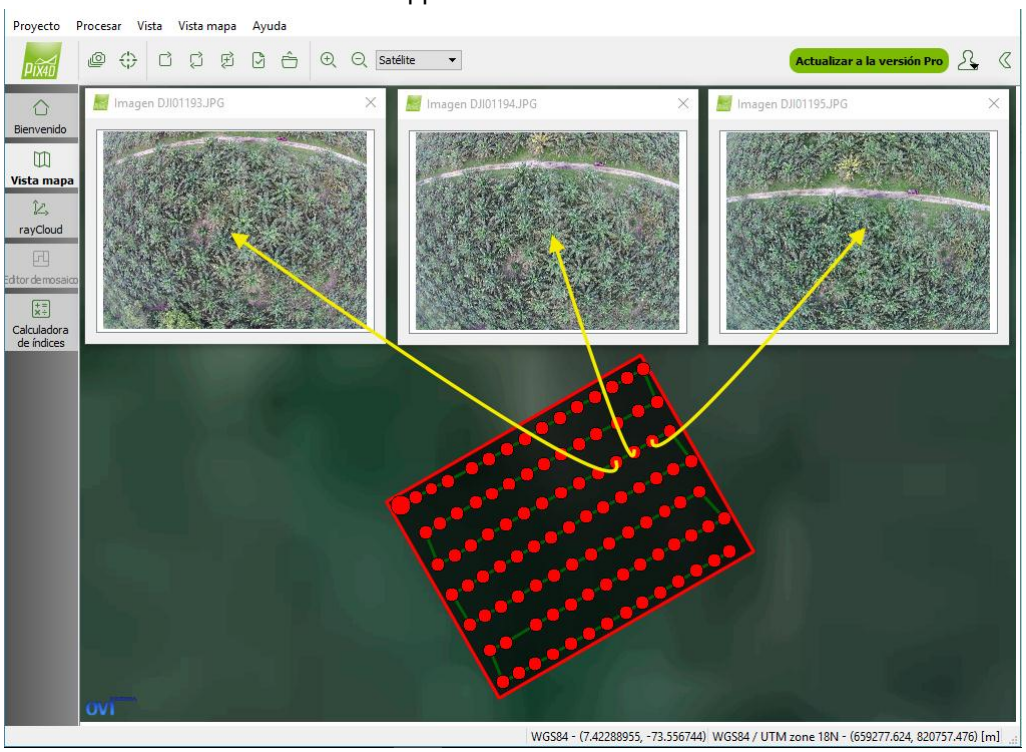
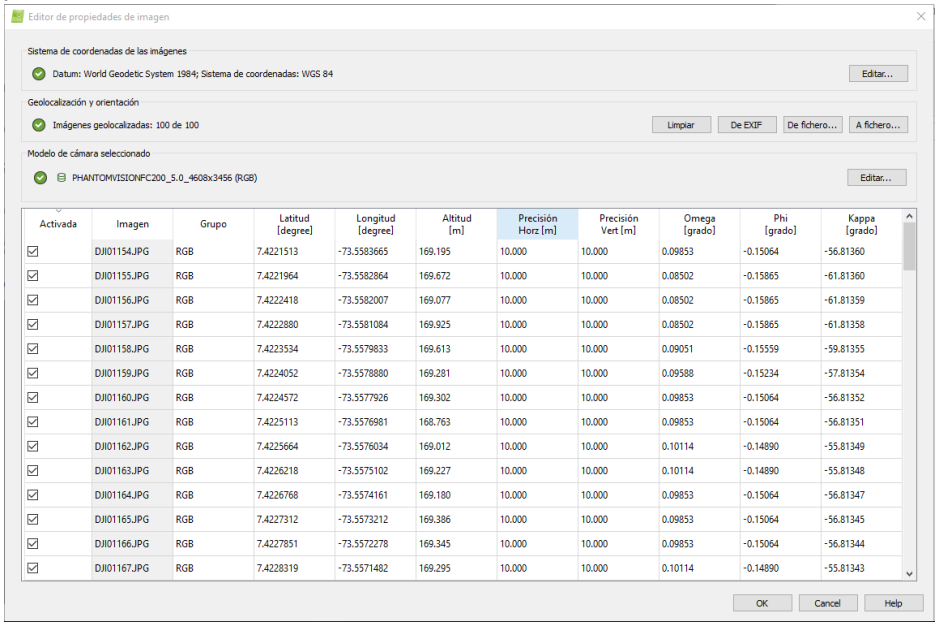


Figura 32. Visualización de los datos recolectados por el UAV en la interfaz del software Pix4Dmapper



Para obtener el ortomosaico correspondiente al plan de vuelo realizado previamente, se debe realizar primero la etapa inicial de procesamiento, para ello se selecciona la opción correspondiente a este proceso y se configuran sus parámetros tal como se muestra en la Figura 33 y en la Figura 34. El tiempo que toma realizar este proceso depende de la cantidad de imágenes que contiene un proyecto de Pix4Dmapper y de la resolución de cada una ellas.

Figura 33. Parámetros de configuración establecidos en la etapa de procesamiento inicial realizada por el software Pix4Dmapper

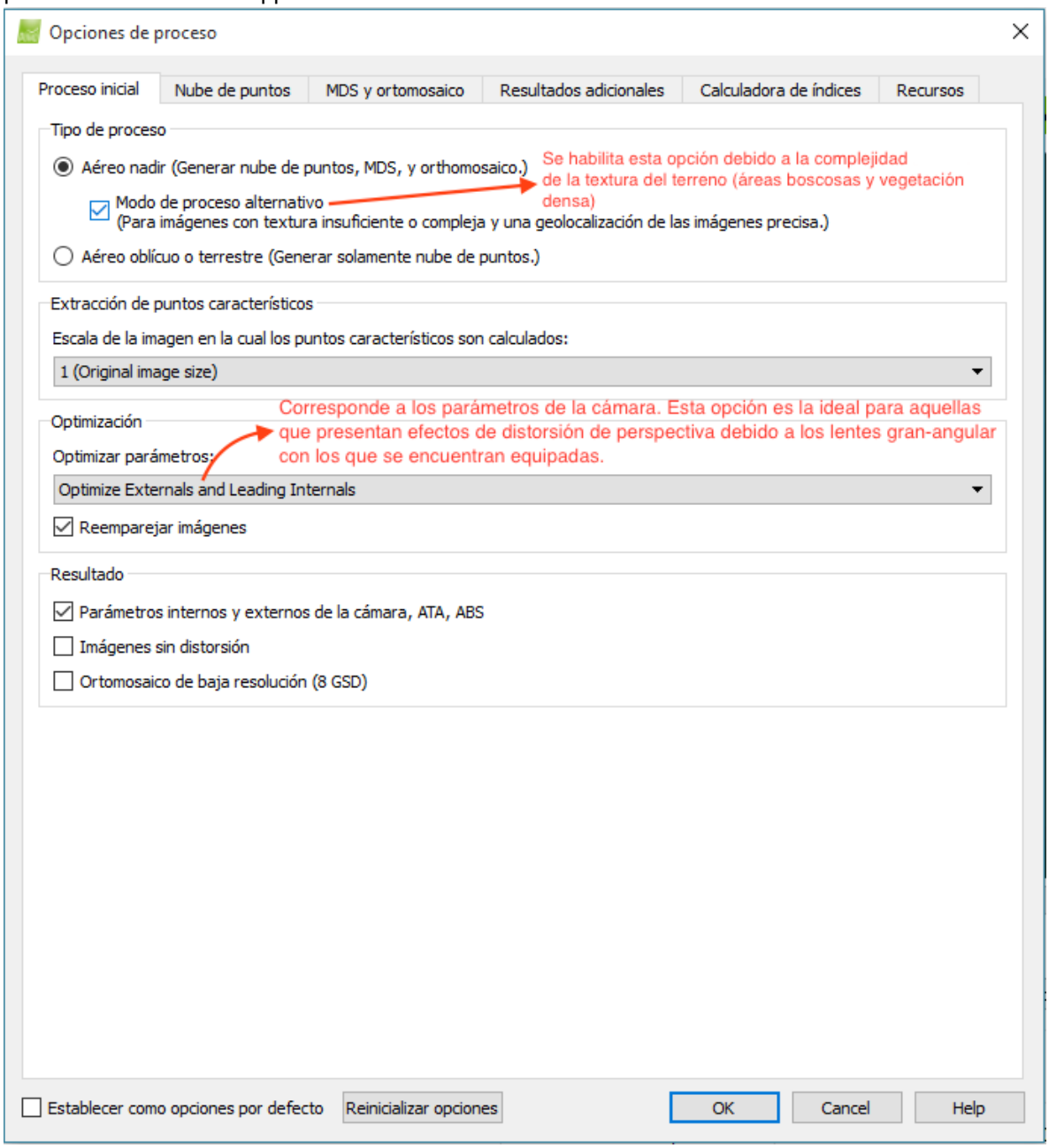
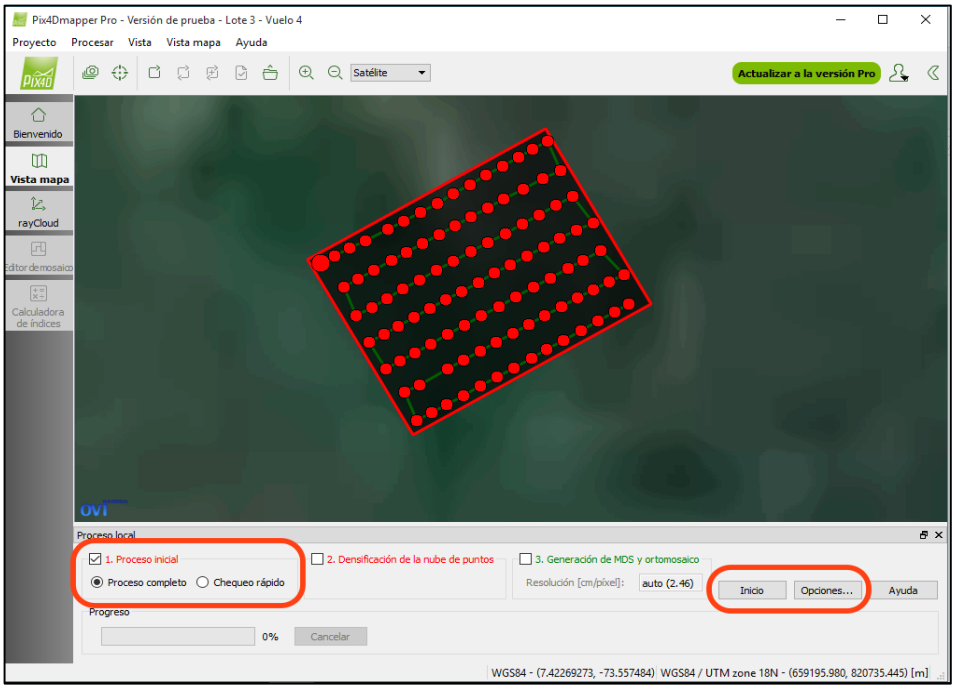
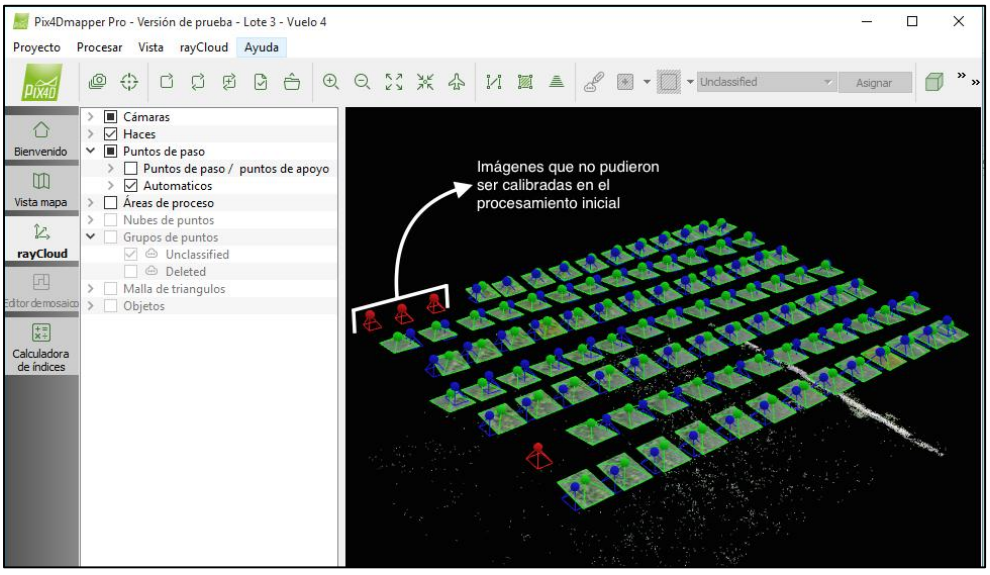


Figura 34. Selección del proceso inicial de procesamiento en el software Pix4Dmapper



Al finalizar la etapa de procesamiento inicial, se obtiene como resultado un modelo con una nube de puntos que contiene cada uno de los *keypoints* que pudieron ser encontrados y emparejados durante este proceso. La Figura 35 permite apreciar este resultado.

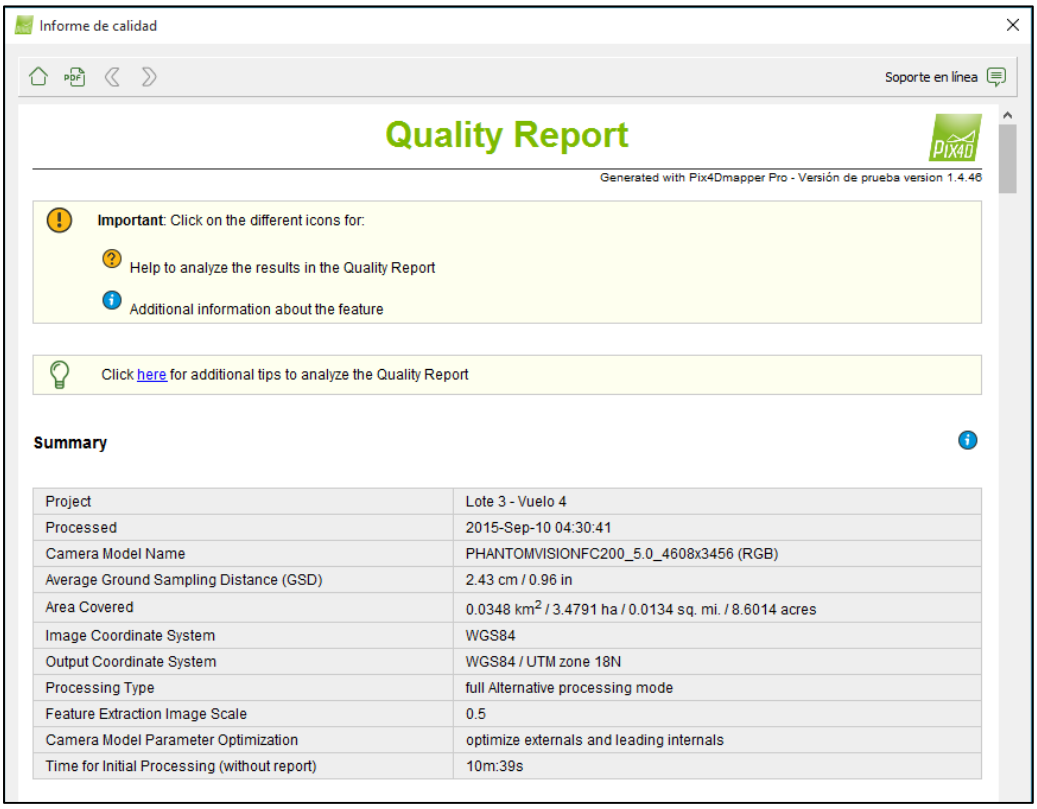
Figura 35. Nube de puntos con los *keypoints* extraídos en la etapa de procesamiento inicial



El software Pix4Dmapper también genera un informe o reporte de calidad para cada proyecto luego de realizar el procesamiento inicial. Teniendo en cuenta estos resultados, los cuales se muestran en la Figura 36 y en la Figura 37, se puede estimar o predecir, si el proyecto cumple o no con las especificaciones técnicas deseadas para generar un ortomosaico con los estándares de calidad que se requieren para que este pueda ser procesado por el software de detección y conteo.

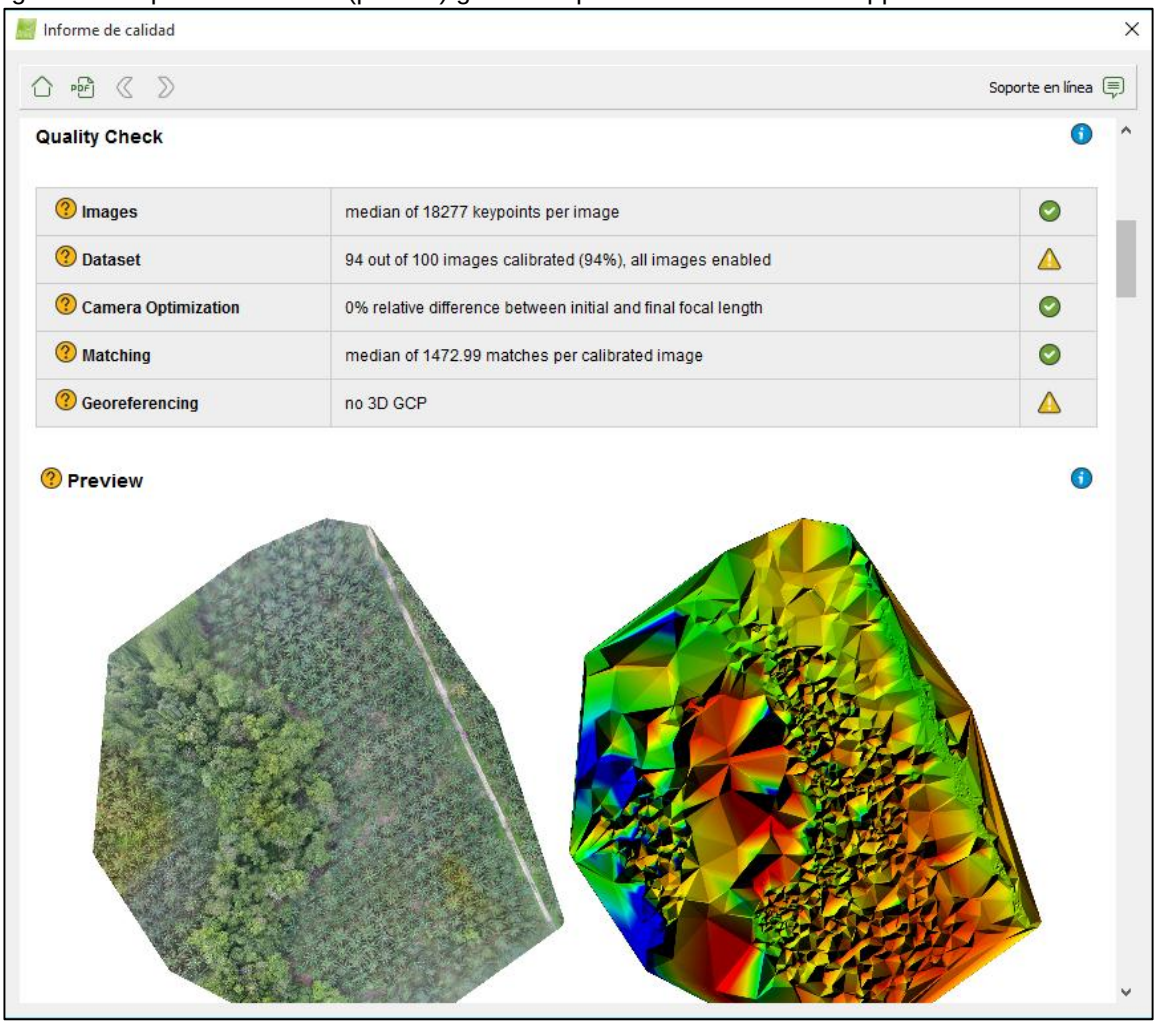
Un proyecto con demasiadas imágenes no calibradas requiere de una gran cantidad de modificaciones, tales como insertar puntos de unión, denominados *'tie points'*, de forma manual en la nube de puntos con el fin de lograr un mayor número de emparejamientos o *matched keypoints*. Este proceso puede ser muy extenuante y agotador cuando se tiene una gran cantidad de imágenes en un proyecto de Pix4Dmapper. En algunas casos un mal resultado puede deberse a configuraciones inadecuadas de los parámetros del programa, por lo que es muy importante estudiar toda la documentación correspondiente al funcionamiento del software.

Figura 36. Reporte de calidad (parte 1) generado por el software Pix4Dmapper



Project	Lote 3 - Vuelo 4
Processed	2015-Sep-10 04:30:41
Camera Model Name	PHANTOMVISIONFC200_5.0_4608x3456 (RGB)
Average Ground Sampling Distance (GSD)	2.43 cm / 0.96 in
Area Covered	0.0348 km^2 / 3.4791 ha / 0.0134 sq. mi. / 8.6014 acres
Image Coordinate System	WGS84
Output Coordinate System	WGS84 / UTM zone 18N
Processing Type	full Alternative processing mode
Feature Extraction Image Scale	0.5
Camera Model Parameter Optimization	optimize externals and leading internals
Time for Initial Processing (without report)	10m:39s

Figura 37. Reporte de calidad (parte 2) generado por el software Pix4Dmapper



8.3.2 Generación de ortomosaicos

El proceso de generación del ortomosaico se inicia luego de haber realizado el procesamiento inicial y haber obtenido resultados satisfactorios en esta etapa. Para generar el ortomosaico se selecciona la opción correspondiente a este proceso como se observa en la Figura 38 y se configuran sus parámetros tal como se muestra en la Figura 39.

Figura 38. Selección del proceso de generación de ortomosaico en el software Pix4Dmapper

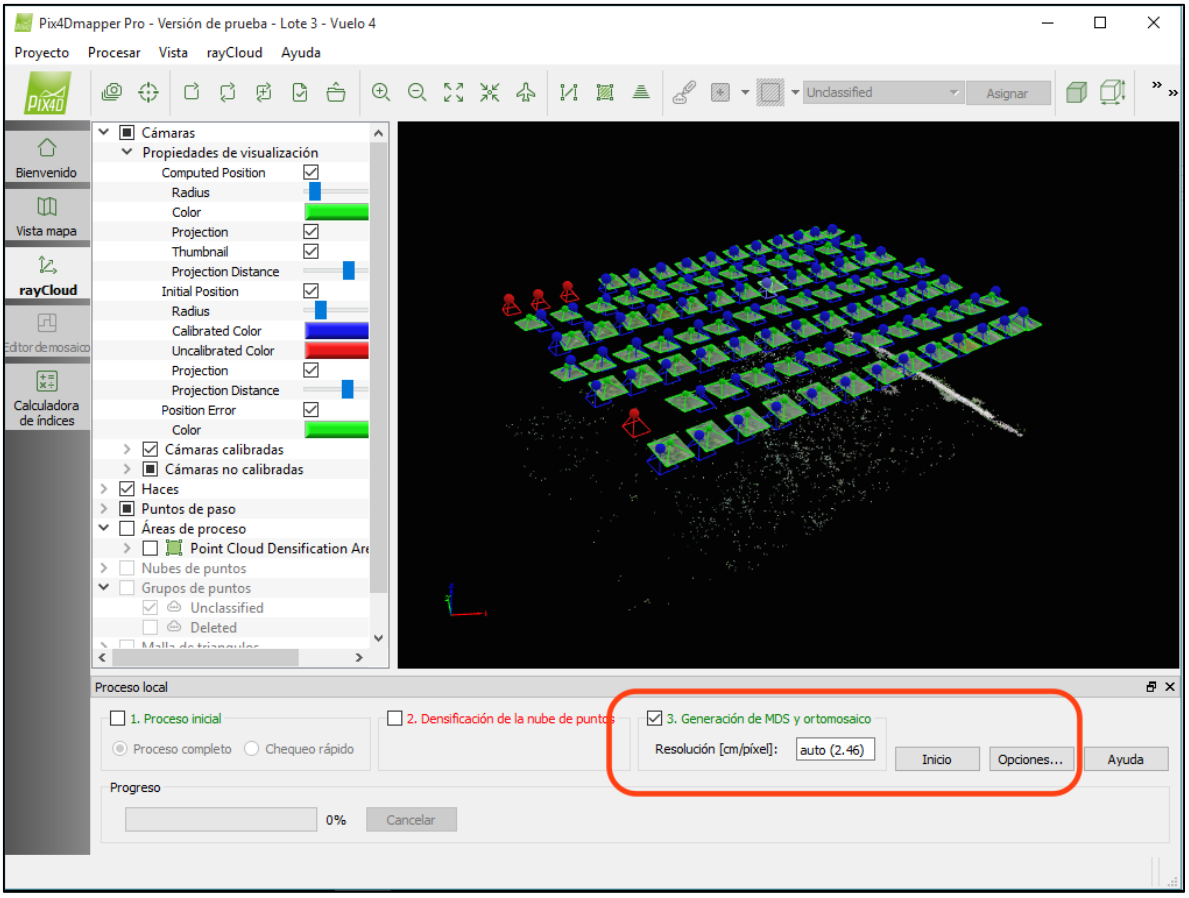
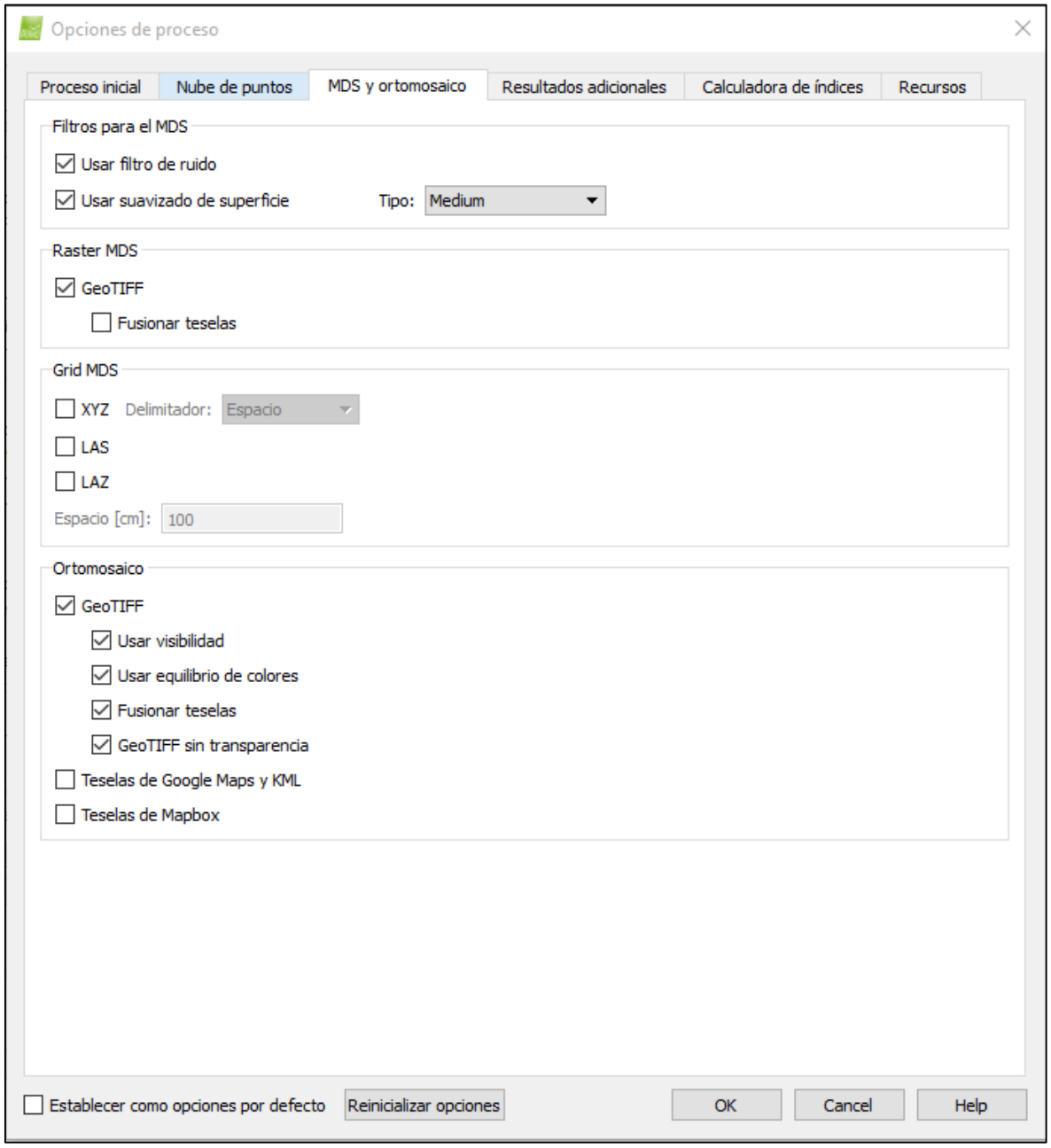


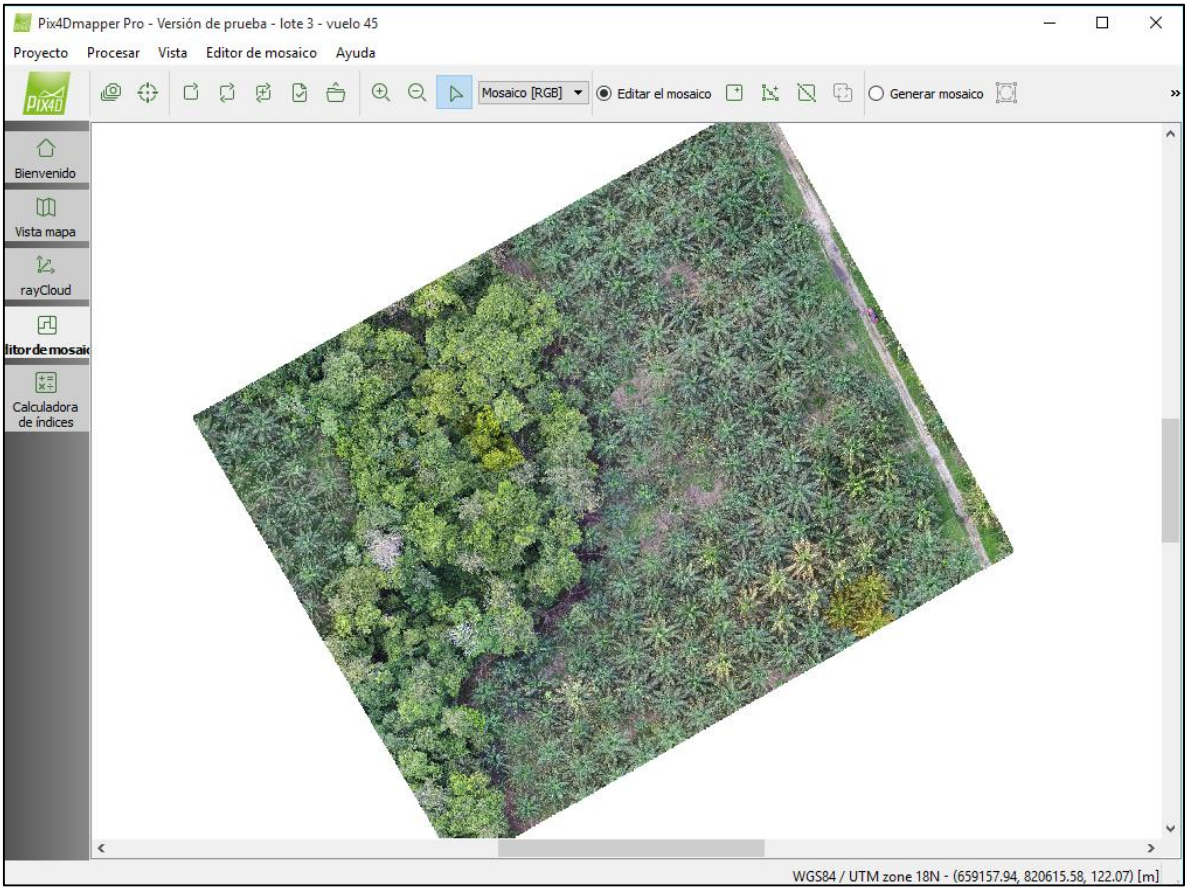
Figura 39. Parámetros de configuración establecidos en el proceso de generación del ortomosaico



Tras finalizar el proceso de creación del ortomosaico, se obtiene una imagen como la que aparece en la Figura 40.

Finalmente todo el procedimiento descrito anteriormente, desde el numeral 8.2.1, se realiza nuevamente en distintas zonas de la plantación con el objetivo de obtener suficientes ortomosaicos para poder continuar con la siguiente fase de este proyecto.

Figura 40. Resultado generado por el software Pix4Dmapper en el proceso de generación de ortomosaico



9. SOFTWARE DE DETECCIÓN Y CONTEO DE PALMAS

Este bloque del sistema de conteo recibe las imágenes que ha generado el software de fotogrametría, de tal forma que cada una de ellas representa un lote o una zona de interés en la plantación. Cada imagen es procesada de forma independiente y por tanto no hay restricción con respecto al número de imágenes que deban ingresarse al software.

Para desarrollar un programa que cuente las palmas en una fotografía de una zona de interés, es necesario que primero sea capaz de reconocer y detectar las palmas que en ella se encuentran, pues el conteo será la sumatoria de estas detecciones.

Es así como se plantea la necesidad de un sistema de detección de palmas que afronte las siguientes tareas:

1. Realizar un recorrido a través de la imagen para tomar en consideración todas las posibles ventanas que puedan contener una palma.
2. Modelar el contenido de cada una de estas ventanas como características que puedan ser procesadas matemáticamente y que permitan discriminar entre la clase “Palma” de la clase “Fondo”.
3. Evaluar cada ventana para decidir si en ella se encuentra o no una palma.

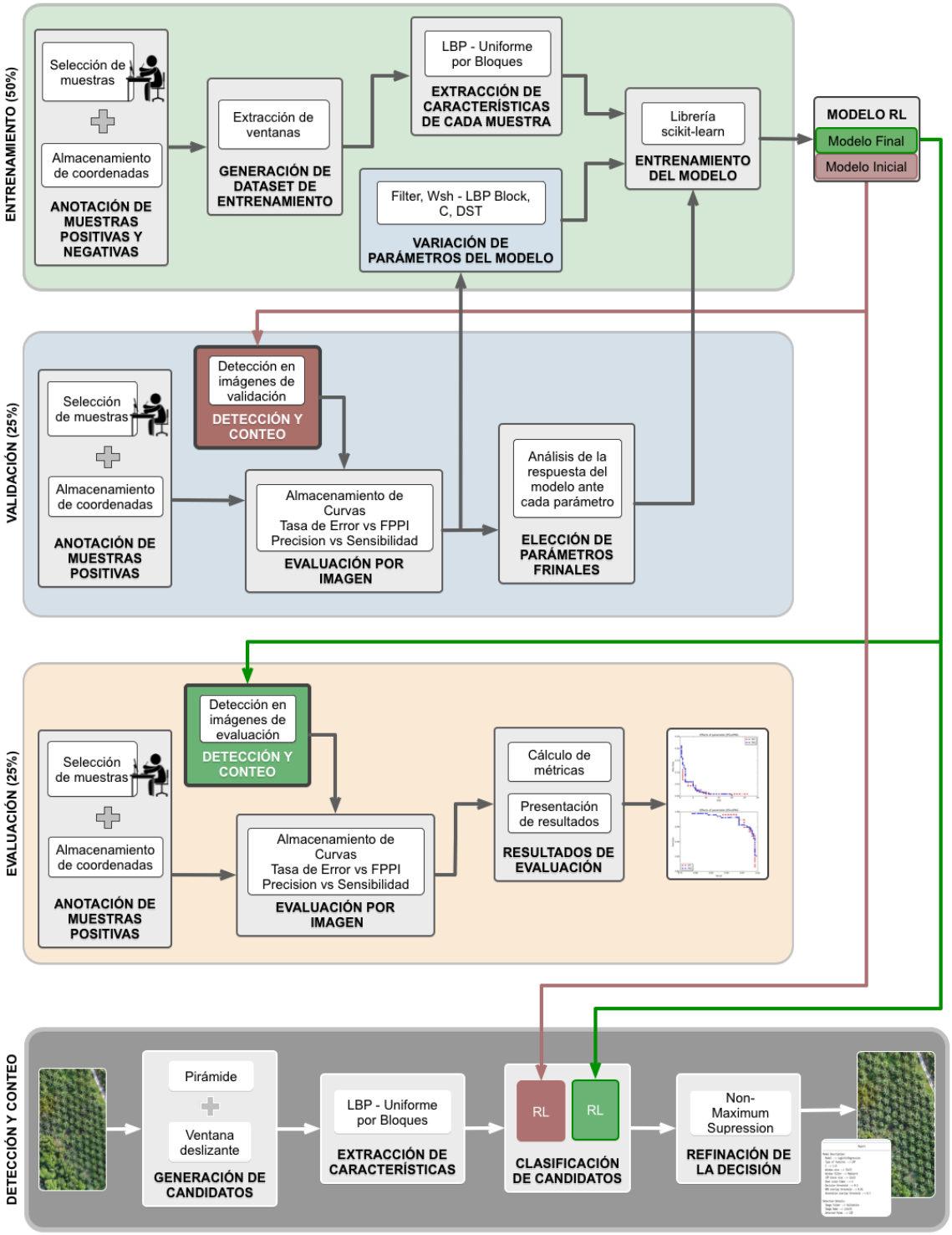
Considerando estas tareas fundamentales, el software de detección y conteo desarrolla la secuencia de procesamiento presentada en el bloque gris “Detección y conteo de palmas en una imagen” de la Figura 41 cada vez que evalúa una imagen de una zona de interés. Este diseño se basa en el esquema básico de un detector de objetos descrito por López, Valveny y Vanrell[56].

Para desarrollar el modelo que decide, para cada una de las ventanas evaluadas, si en ella se localiza o no una palma, se requieren 3 etapas previas de desarrollo.

En la primera de ellas el modelo es entrenado con un grupo de imágenes de forma que realiza un proceso de aprendizaje para reconocer las características propias de cada clase, obteniendo un modelo preliminar o por defecto.

[56] LÓPEZ, Antonio; VALVENY, Ernest y VANRELL, María. Detección de objetos.[Diapositivas] En: Detección de Objetos [Curso online]. Universitat Autònoma de Barcelona: mayo-julio, 2015. Disponible en: Coursera <https://www.coursera.org/course/deteccionobjetos>

Figura 41. Composición del sistema de conteo: software de detección y etapas de entrenamiento, validación y evaluación



Posteriormente, otro grupo de imágenes es utilizada para la validación del modelo, en la cual se evalúan las imágenes y basados en esos resultados, se varían y se ajustan los parámetros del modelo para optimizar su desempeño en un grupo distinto al de entrenamiento, evitando que se sobreajuste a las muestras iniciales y dando como resultado un modelo final.

Finalmente, el sistema es evaluado en un grupo de imágenes con las que no había interactuado antes con el fin de obtener métricas de evaluación que generalicen el rendimiento del modelo final que será utilizado para la detección y conteo de palmas.

9.1 DISTRIBUCIÓN DE LOS CONJUNTOS DE IMÁGENES

Para el desarrollo de todo el proyecto se realizaron jornadas de adquisición de imágenes en campo con el UAV, las cuales fueron procesadas posteriormente con la herramienta de fotogrametría. Se obtuvo un total de 8 ortomosaicos que contienen más de 1600 palmas.

Se busca entrenar el modelo con la mayor cantidad de muestras posibles para que tenga un amplio conocimiento de todas las diferentes palmas y fondos que pueda encontrar, y por tanto podría suponerse que se usarían todas las muestras disponibles para este proceso. Sin embargo, la predicción que hace el modelo para las muestras con las cuales es entrenado no da información acerca de su capacidad de generalización, existiendo la posibilidad de que se sobreajuste solamente a ese grupo.

Para solucionar esto, Ng[57] propone que la base de datos mantenga un grupo de imágenes que no se involucren en el entrenamiento. Sin embargo, cuando los parámetros vuelven a ajustarse para obtener la mejor calificación posible en esta evaluación, se considera que este nuevo porcentaje ya se ha involucrado en el proceso de obtención del modelo final y por tanto no se considera correcta una generalización a partir del mismo. Es por esto que nace la necesidad de un tercer grupo de muestras, que no son usadas en el entrenamiento y que tampoco inciden en los ajustes realizados al modelo, sobre las cuales se obtiene una calificación que sí refiere a la capacidad de generalización con muestras desconocidas.

[57] NG, Andrew. Model selection and Train/Validation/Test sets. Advice for applying machine learning. Machine learning [Curso online]. Stanford University, 2015. Disponible en: Coursera <https://www.coursera.org/learn/machine-learning/ >

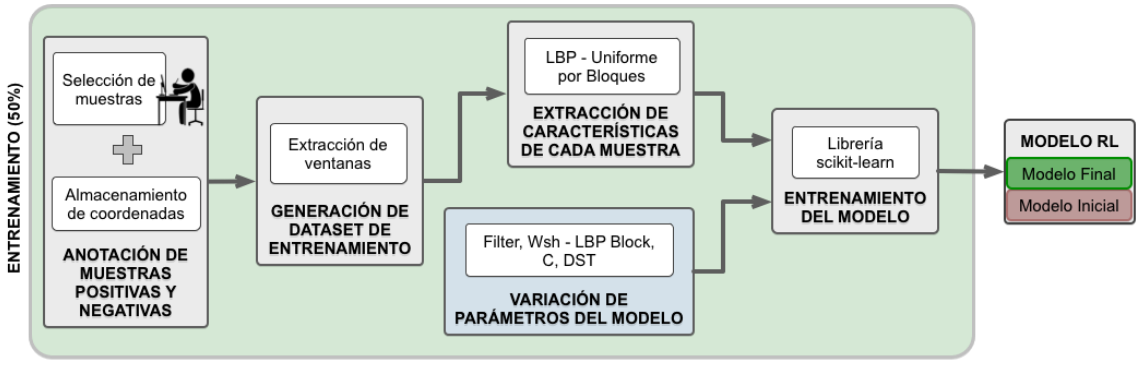
Ng recomienda que la base de datos obtenida sea distribuida en 3 grupos: el 60% de las imágenes como conjunto de entrenamiento, el 20% como grupo de validación donde se realizan las primeras evaluaciones del modelo y se reajustan los parámetros para maximizar el rendimiento, y finalmente, el 20% restante asignado a un grupo de evaluación final que no realimenta al modelo sino que generaliza su desempeño.

Siguiendo un aproximado de esta asignación y procurando que el número de ortomosaicos se divida de forma entera, la base de datos obtenida para este proyecto se distribuye así: 4 ortomosaicos para entrenamiento (50%), 2 para validación (25%) y los 2 restantes para evaluación (25%). El porcentaje asignado en cada etapa es presentado en la Figura 41 acompañando el título de cada proceso.

9.2 ENTRENAMIENTO DEL MODELO DE APRENDIZAJE AUTOMÁTICO

Esta fase tiene como objetivo el entrenamiento de un modelo de regresión logística capaz de clasificar ventanas como palmas o fondo. La Figura 42 muestra el diagrama de bloques que describe los pasos de este proceso. La primera implementación de esta etapa permite obtener un modelo preliminar que pasará a un proceso de validación, donde se evaluará su rendimiento ante la variación de sus parámetros. Cada vez que se re ajusten los parámetros del modelo, será necesario reentrenarlo usando este mismo esquema.

Figura 42. Proceso de entrenamiento del modelo de aprendizaje automático



9.2.1 Anotación de muestras positivas y negativas

El proceso de anotación empieza con una selección manual de las muestras positivas (Palmas) y negativas (Fondo) realizadas por una persona.

Inicialmente, se especifican los ajustes de configuración que se observan en la Figura 43. Se establece el nombre de la carpeta donde se encuentra el grupo de imágenes destinadas al proceso de entrenamiento, lo que es necesario, ya que en los procesos de validación y evaluación también se realizan anotaciones usando el mismo programa. Así mismo, se establece el nombre de la imagen (ortomosaico) en el cual se van a identificar las muestras puesto que esta selección se realiza de forma individual para cada imagen.

El proceso de anotación se desarrolla de la siguiente forma:

1. Se carga la imagen en la cual se desean extraer las anotaciones a partir de los datos de carpeta y nombre de imagen dados en las configuraciones del programa.

2. Si la imagen ya tiene anotaciones previas, es decir, ya existe un archivo de texto en la misma carpeta identificado con el nombre que lo asocia a esta imagen, sus ventanas son cargadas.

Figura 43. Configuraciones para el programa de selección de muestras

```
#############################################################################
###                          CONFIGURATIONS                               ###
#############################################################################

#Image folder must be 'Training', 'Validation' or 'Test'
imageFolder = 'Training'

#The name of the image which you want to make annotation file
imageName = 'Lote3B'

#Category: 'Palm' or 'Background'
category = 'Palm'

#Save or not the annotation file: True or False
save = True

#############################################################################
```

3. Se abre la imagen en una ventana que permite la interacción con el usuario. Cuando existen anotaciones previas, estos recuadros anteriores son cargados en color azul.

La Figura 44 presenta la ventana de interacción que se carga para la imagen solicitada en las configuraciones de la Figura 43, cuando la categoría seleccionada es "*Palm*" (Palma), mientras que la Figura 45 presenta como lucen estas ventanas cuando se realizan las anotaciones de las muestras negativas seleccionando la categoría "*Background*" (Fondo).

4. La persona que realiza las anotaciones da click sobre cada punto central donde considera que se debe localizar una ventana.

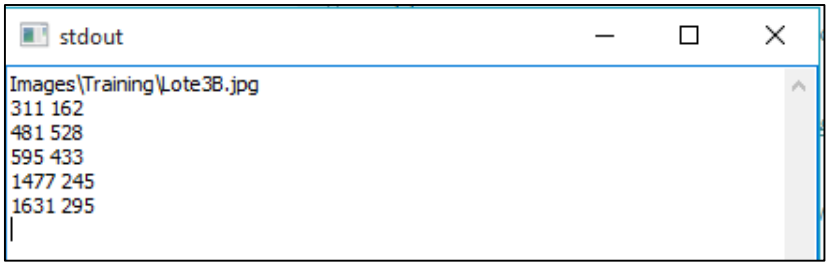
5. El programa dibuja una ventana con el tamaño que ha sido configurado en el archivo de configuraciones general del proyecto. En este caso, se ha decidido iniciar el modelo preliminar con una ventana de 72x72 pixeles en la cual puede enmarcarse el centro de una palma grande sin incluir todas sus ramificaciones. Estas nuevas ventanas anotadas se observan en la Figura 44 y en la Figura 45 de color verde.

6. Una vez que el usuario complete todas las anotaciones deseadas y cierre la ventana de interacción, la consola reportará las coordenadas centrales de cada recuadro que ha sido anotado en esta ejecución, como se observa en la Figura 46 .

Figura 44. Interfaz de anotaciones con ventanas para categoría Palma

Figura 45. Interfaz de anotaciones con ventanas para categoría Fondo

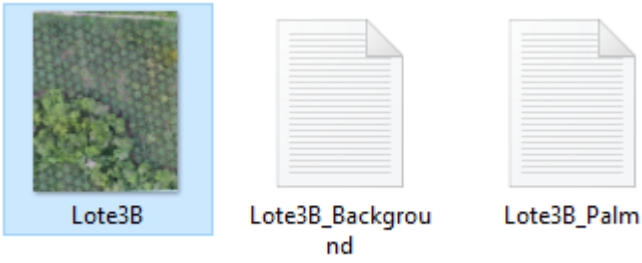
Figura 46. Respuesta de la consola con las nuevas coordenadas anotadas



7. Si la opción de guardar ha sido activada en las configuraciones mediante “save = True”, estas coordenadas serán almacenadas en un archivo de texto con extensión .txt, almacenado en la misma carpeta en donde se encuentra la imagen. El nombre de este archivo tendrá la estructura: “*nombre de la imagen_categoría.txt”.* La Figura 47 presenta cómo luce, desde el explorador de archivos, la imagen junto a sus anotaciones de palmas y fondos. En el caso, en que este archivo de texto ya existiese porque ya se hubiera realizado una anotación previa, las nuevas coordenadas serán añadidas al mismo.

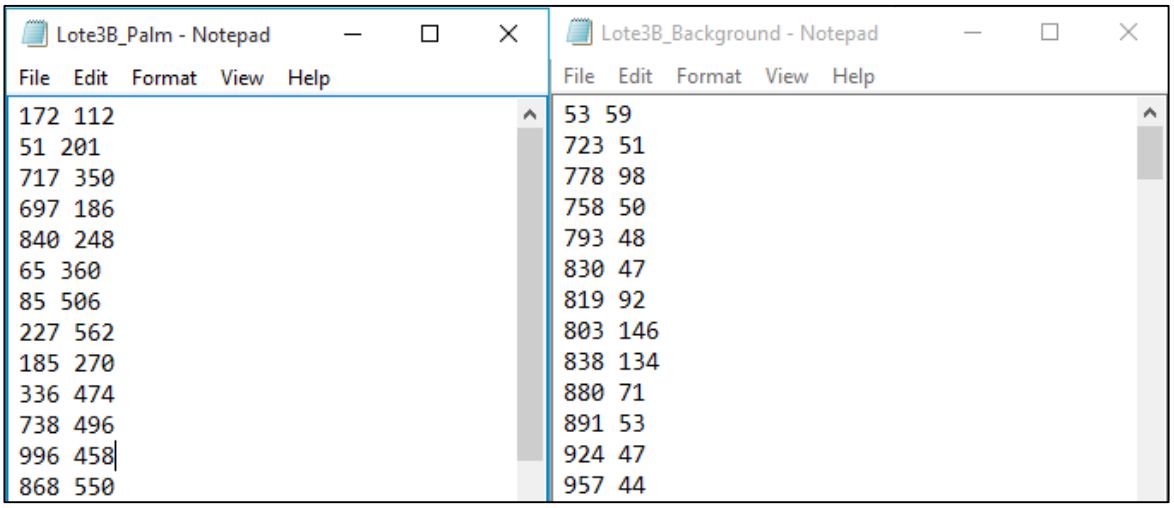
   Si la opción de guardar no ha sido activada, la consolará reportará las coordenadas centrales indicadas por la persona en esta sesión pero no escribirá ni modificará los archivos de texto.

Figura 47. Imagen con sus respectivos archivos de texto conteniendo las coordenadas de palmas y fondos



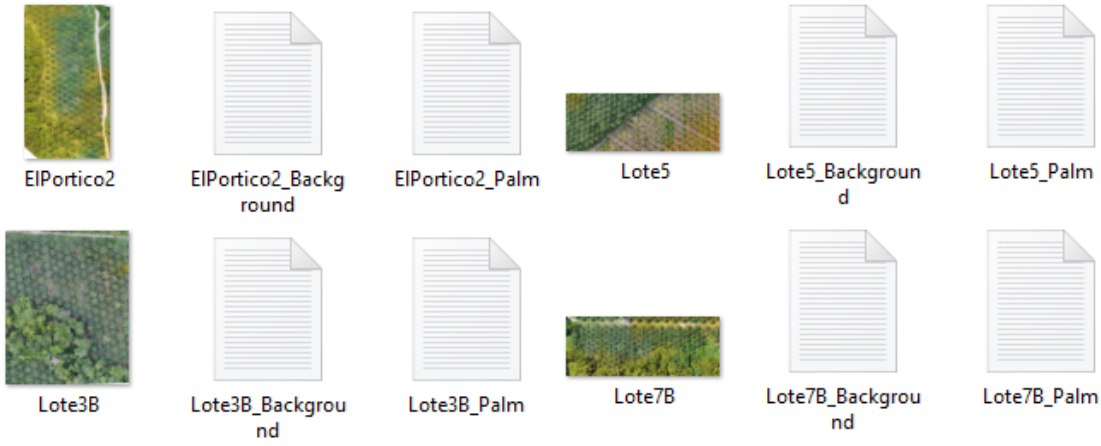
Como resultado de este proceso, se obtienen 2 archivos de texto para cada imagen, como los que presenta la Figura 48. Cada uno de ellos contendrá las coordenadas (x_{centro}, y_{centro}) de las ventanas indicadas en cada categoría.

Figura 48. Archivos de texto con las coordenadas centrales de las ventanas anotadas para un ortomosaico



Este proceso de anotación descrito para una imagen, es realizado con todos los ortomosaicos que conformen el grupo de entrenamiento. Como resultado, se tendrá una carpeta que contendrá todas las imágenes de entrenamiento y sus respectivas anotaciones, como se observa en la Figura 49.

Figura 49. Ortomosaicos de entrenamiento con sus respectivos archivos de anotaciones



9.2.2 Generación de base de datos de entrenamiento

Cuando el sistema busca las posibles palmas dentro de una imagen, genera diversas ventanas candidatas y cada una de ellas recibe una calificación por parte del modelo de aprendizaje automático para establecer la categoría a la que pertenece.

El modelo utilizado pertenece a la categoría de aprendizaje supervisado y por tanto requiere la existencia de etiquetas, es decir, de datos cuya categorización correcta sea conocida. En este caso, los datos refieren a ventanas, por lo que el modelo necesita para su entrenamiento una base de datos de imágenes con el tamaño de las ventanas que va a evaluar y con la indicación adecuada de la categoría correcta.

El proceso de anotación descrito previamente permite identificar las coordenadas donde se centran las ventanas de palmas y fondo. El bloque de generación de base de datos se basa en dichas coordenadas para extraer las ventanas positivas y negativas indicadas para los ortomosaicos en la carpeta de entrenamiento.

Para esto, se listan inicialmente todos los archivos contenidos en la carpeta de imágenes de entrenamiento. Allí se filtran aquellos con formato de imagen para conocer exactamente el nombre de cada ortomosaico y para cada uno de ellos se buscan los archivos de texto con las anotaciones de ventanas. En caso de que estos no existan, surge un error del cual el programa informa.

En el caso en el que los archivos de texto correspondientes existan, el programa los abre y extrae la ventana correspondiente a cada coordenada. El proceso se

realiza primero para las muestras positivas, y posteriormente se repite con las muestras negativas.

Para la extracción de la ventana se hace uso de la herramienta "*crop*" incluida en la librería PIL (*Python Imaging Library*) de Python, la cual recorta un rectángulo de una imagen al recibir la coordenada (x_1, y_1) correspondiente a la esquina superior izquierda y la coordenada (x_2, y_2) correspondiente a la esquina inferior derecha.

La Figura 50 presenta el código que permite implementar esta herramienta y la conversión de coordenadas requerida a partir de la coordenada central indicada en las anotaciones. El tamaño de la ventana es extraído de las configuraciones generales del proyecto, de esta forma, el programa funciona para extraer cualquier tamaño de ventana especificado.

Figura 50. Conversión de coordenadas y extracción de ventana con herramienta "*crop*"

```
center = (int(x),int(y))

x1 = int(center[0] - windowSize[0]/2)
y1 = int(center[1] - windowSize[1]/2)
x2 = int(center[0] + windowSize[0]/2)
y2 = int(center[1] + windowSize[1]/2)

box = (x1,y1,x2,y2)
cropImage = image.crop(box)
```

Como resultado de este bloque, se obtiene una base de datos de entrenamiento con ventanas de tamaño específico (72x72 pixeles) divididas en 2 categorías: 450 muestras positivas y 2500 muestras negativas, para una base de datos compuesta por un total de 2950 muestras. La Figura 51 presenta una muestra de las ventanas extraídas para la categoría Palma, mientras que en la Figura 52 se observan algunas de las extraídas para la categoría Fondo.

Figura 51. Muestras positivas para entrenamiento

Figura 52. Muestras negativas para entrenamiento

9.2.3 Extracción de características

Para que el modelo pueda diferenciar entre ventanas, es necesario que cada una de ellas sea descrita por medio de características numéricas que den información acerca de su composición. De esta forma se podrá establecer una diferenciación entre las características de una ventana con una palma y una con fondo. Este bloque obtiene las características que representan a cada ventana de entrenamiento mediante el uso del operador LBP (*Local Binary Pattern*).

El proceso de extracción de características es realizado de forma individual para cada ventana dentro de la base de datos e inicia convirtiéndola de RGB a escala de grises. Esto permite que la ventana sea representada en una matriz de dos dimensiones como la presentada en la Figura 53, en la cual se da información del nivel de luminancia de cada pixel basándose en la imagen RGB original, pero sin tener que manejar una matriz de 3 dimensiones.

La Figura 54 presenta un ejemplo de ventana de la categoría Palma cuando es transformada a escala de grises, junto a un ejemplo de la categoría Fondo que pasa por la misma transformación. La categoría de Fondo incluye todo aquello que no es el centro de una palma, como caminos, casas, carros o árboles, por tanto el ejemplo seleccionado es uno de los posibles fondos diversos que puede analizar, sin embargo, corresponde a uno de los más difíciles de diferenciar pues presenta las ramas de las palmas que no definen el centro de una palma pero cuyo patrón se parece bastante al de la categoría Palma.

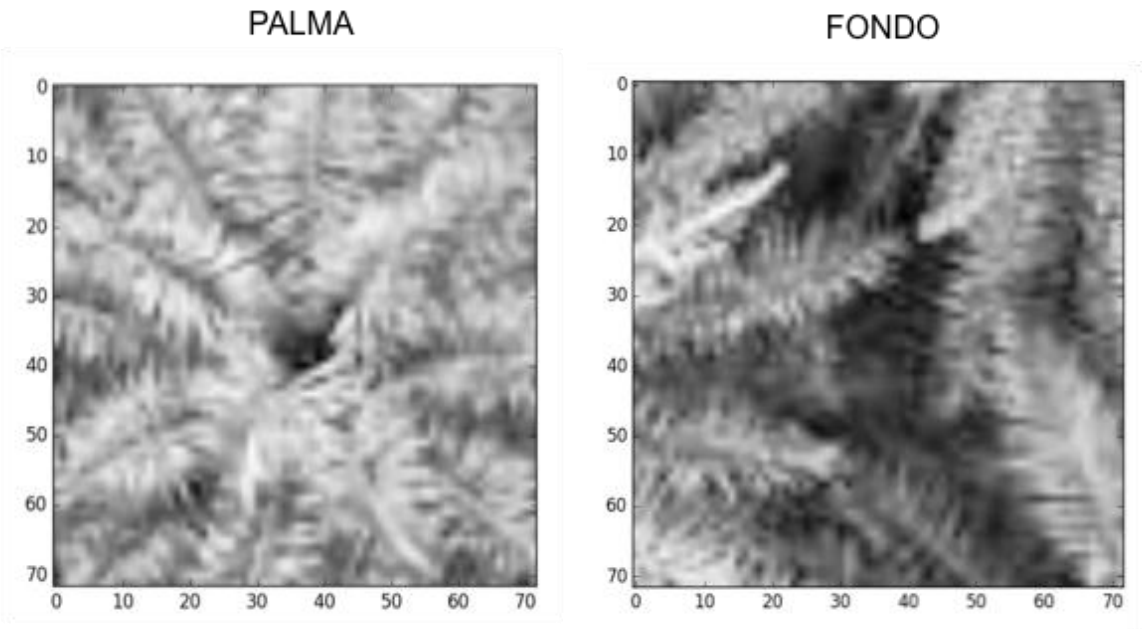
Figura 53. Imagen en escala de grises como matriz de dos dimensiones

Python Variable Browser

Name	Type	Value
image	uint8 [72x72] ndarray	[[149 116 148 ..., 142 152 168] [112 105 142 ..., 15...
imageFolder	str [8]	Training

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	149	116	148	94	101	98	104	111	108	100	132	158	137	127	144	151	171	155
1	112	105	142	93	108	108	106	108	136	144	150	152	144	130	135	156	151	167
2	98	113	144	97	116	111	99	103	103	138	147	148	160	142	122	135	132	159
3	127	142	154	114	125	103	83	98	116	133	126	116	129	137	132	137	130	142
4	158	160	160	139	141	97	73	96	142	148	165	172	168	172	154	112	128	143
5	161	155	153	157	151	103	81	101	136	124	134	139	134	157	163	127	115	147
6	157	146	142	156	146	115	104	113	109	126	149	166	177	188	180	159	120	129
7	161	147	136	146	132	121	124	123	118	151	157	159	180	181	169	178	139	101
8	166	177	122	127	155	133	127	108	121	139	126	103	163	172	164	159	148	140
9	183	187	134	140	165	146	144	133	159	148	137	136	179	157	135	129	138	142
10	189	173	137	169	188	144	136	146	121	117	131	142	157	140	145	155	149	158
11	131	129	114	145	171	156	157	160	106	116	142	147	140	144	157	150	145	155
12	114	139	130	121	147	181	189	163	143	134	145	159	156	169	159	125	126	135
13	118	119	124	139	158	169	165	152	163	140	131	144	129	141	146	143	149	145

Figura 54. Ventanas de Palma y Fondo en escala de grises



Se hace uso de la herramienta "*local_binary_pattern*" incluida en la librería scikit-image para realizar la extracción de las características LBP de la ventana en escala de grises. Se utiliza una vecindad de 8 pixeles y se emplea la variante LPB-Uniforme para reducir el número de posibles patrones, de forma que cada pixel toma un valor entre [0-58] de acuerdo al patrón de textura que lo rodea.

Como resultado del descriptor LBP-U, la nueva matriz de cada ventana es una descripción de la textura contenida en la imagen original. Si se tomase directamente esta matriz como la representación matemática se tendría un vector con una dimensión de 5184 valores (número total de pixeles), por tanto es preferible usar el histograma normalizado de esta imagen, pues contiene la información acerca de la cantidad de pixeles que presentan cada patrón LBP-U mientras reduce la dimensión del vector de características a 59 valores (los posibles patrones LBP-U). La Figura 55 muestra las imágenes LBP-U obtenidas para las ventanas de palma y fondo de la Figura 54, las cuales son descritas con sus respectivos histogramas en la Figura 56.

Figura 55. Imágenes LBP-U para las ventanas de Palma y Fondo

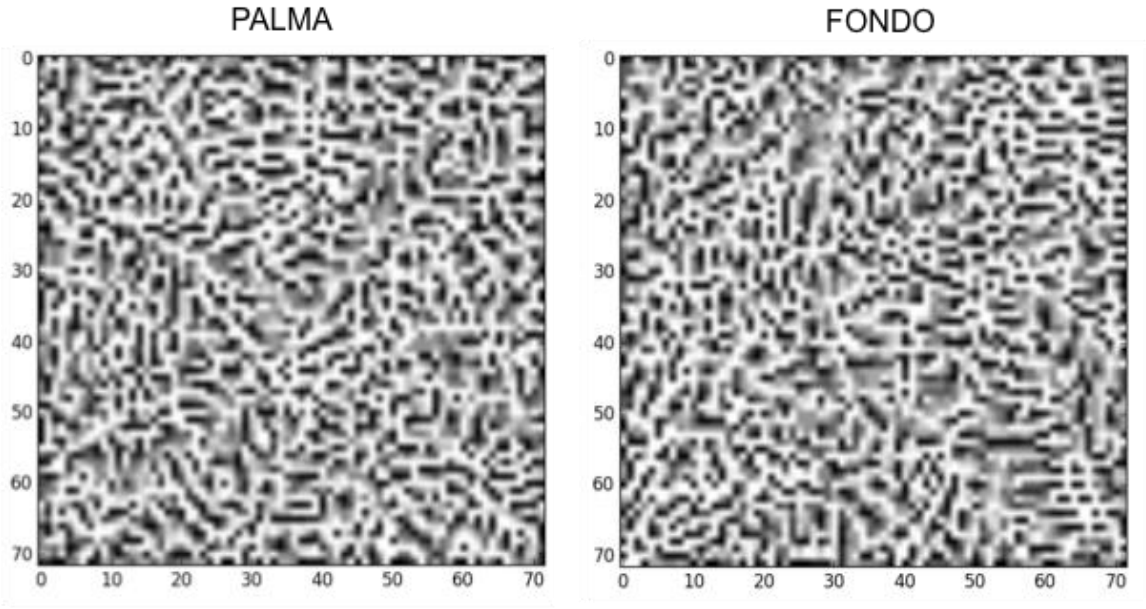
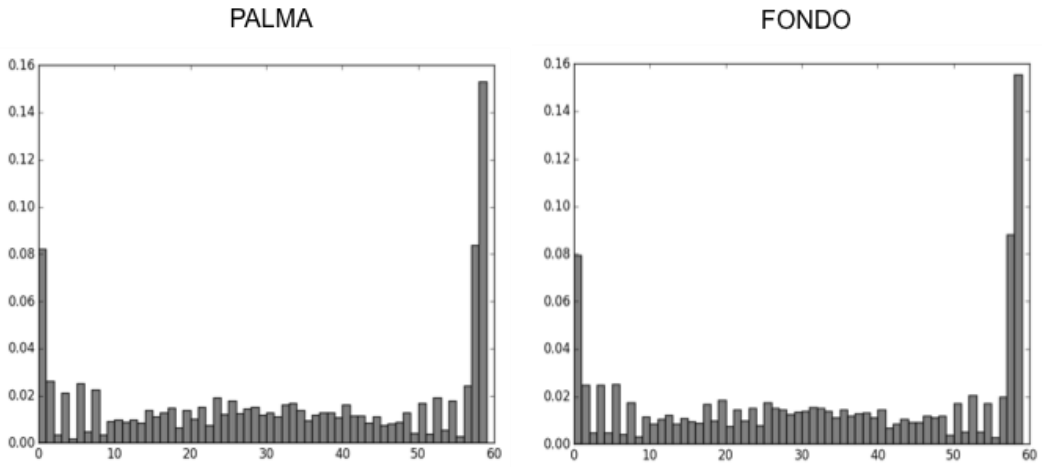


Figura 56. Histogramas de las imágenes LBP-U para las ventanas de Palma y Fondo

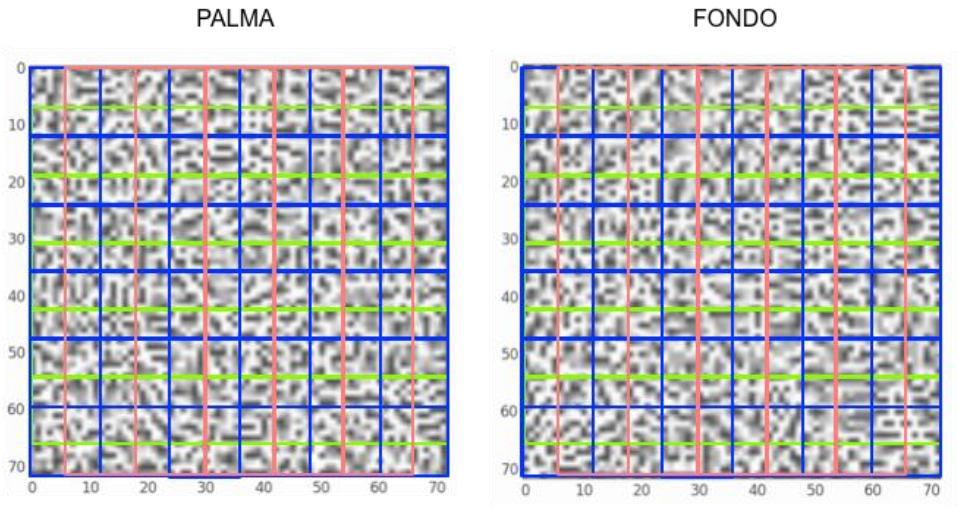


Sin embargo, el histograma LBP-U de una ventana completa puede ser exactamente el mismo aún si se presentan variaciones locales dentro de la misma, ya que el LBP-U es invariante a la traslación. Por tanto, podría representar un histograma muy similar para una ventana colocada en el centro de una palma, como para una que contiene la parte inferior de una palma y la superior de otra,

haciendo que sea más posible que las confunda con los fondos “difíciles” entre palmas.

Para hacer que el LBP-U sea más robusto a estas variaciones, la ventana es dividida en varios bloques que se solapan como se presenta en la Figura 57. Esta organización es realizada mediante la herramienta “*view_as_windows*” de la librería scikit-image, la cual permite subdividir una imagen en ventanas con un tamaño y paso determinado. Para el modelo inicial se ajusta un bloque LBP de 12x12 pixeles con un paso igual a la mitad de este bloque, de forma que cada bloque se solapa con el siguiente en 6 pixeles. El resultado de esta herramienta es una matriz con dimensiones (11x11x12x12), lo que puede interpretarse como una matriz de 11x11 que contiene en cada posición un bloque LBP de tamaño 12x12, obteniendo un total de 121 bloques LBP para una ventana de 72x72.

Figura 57. Distribución de bloques LBP-U en una ventana



Cada bloque LBP-U es representado por su histograma, que tiene 59 posibles patrones, y la ventana completa es caracterizada matemáticamente mediante la secuencia de estos histogramas, como se muestra en la Figura 58. De esta forma, el vector de características de una ventana tiene una dimensión de 7139 valores (121 bloques LBP x 59 características cada uno).

Este vector de características es extraído para cada una de las ventanas que componen la base de datos y es almacenado en un archivo con extensión “.feat” para cada una de ellas. La Figura 59 presenta un ejemplo de los archivos de características para palmas y fondos visualizados desde el explorador de archivos.

Figura 58. Histogramas de bloques LBP-U para representar una ventana

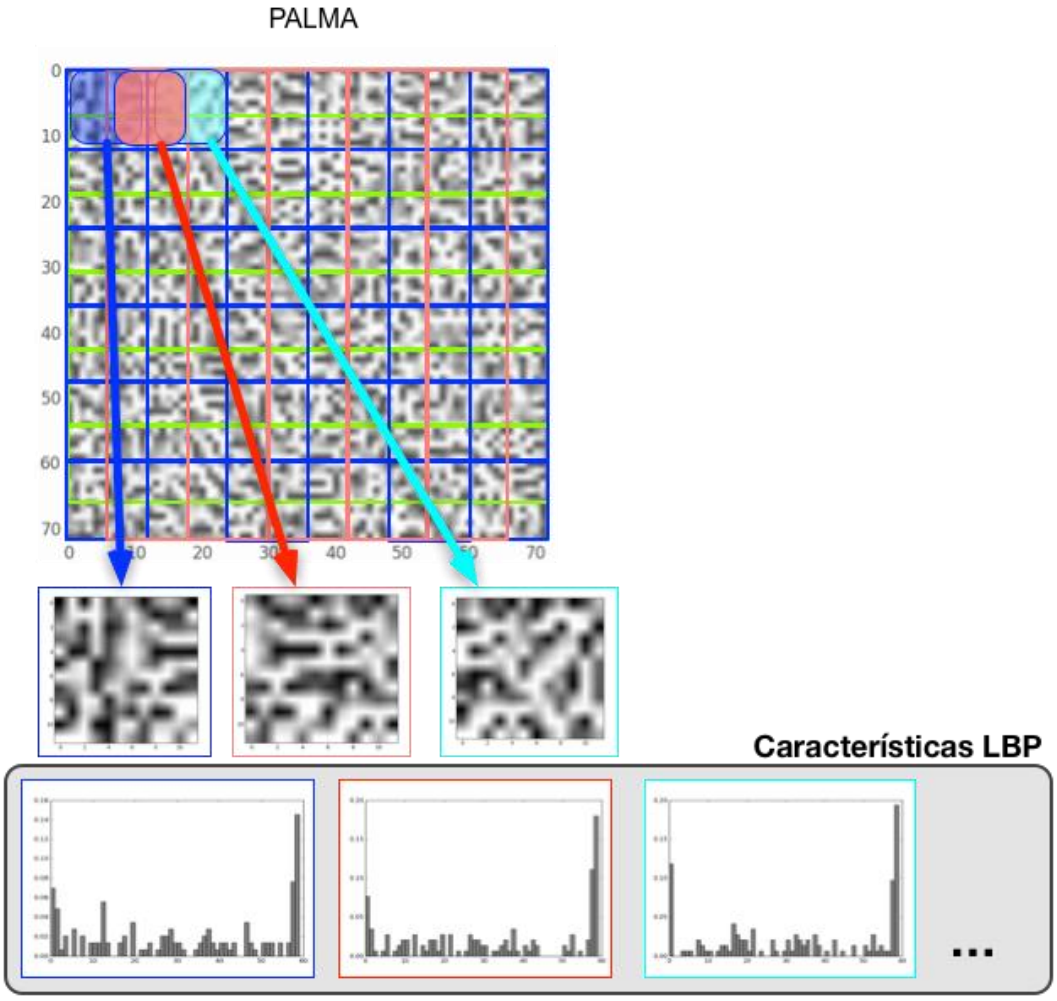
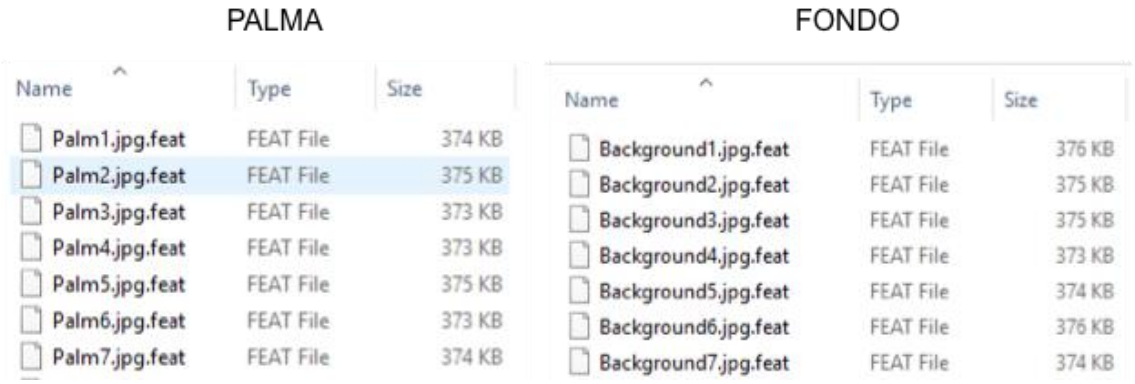


Figura 59. Archivos ".feat" con las características para palmas y fondos

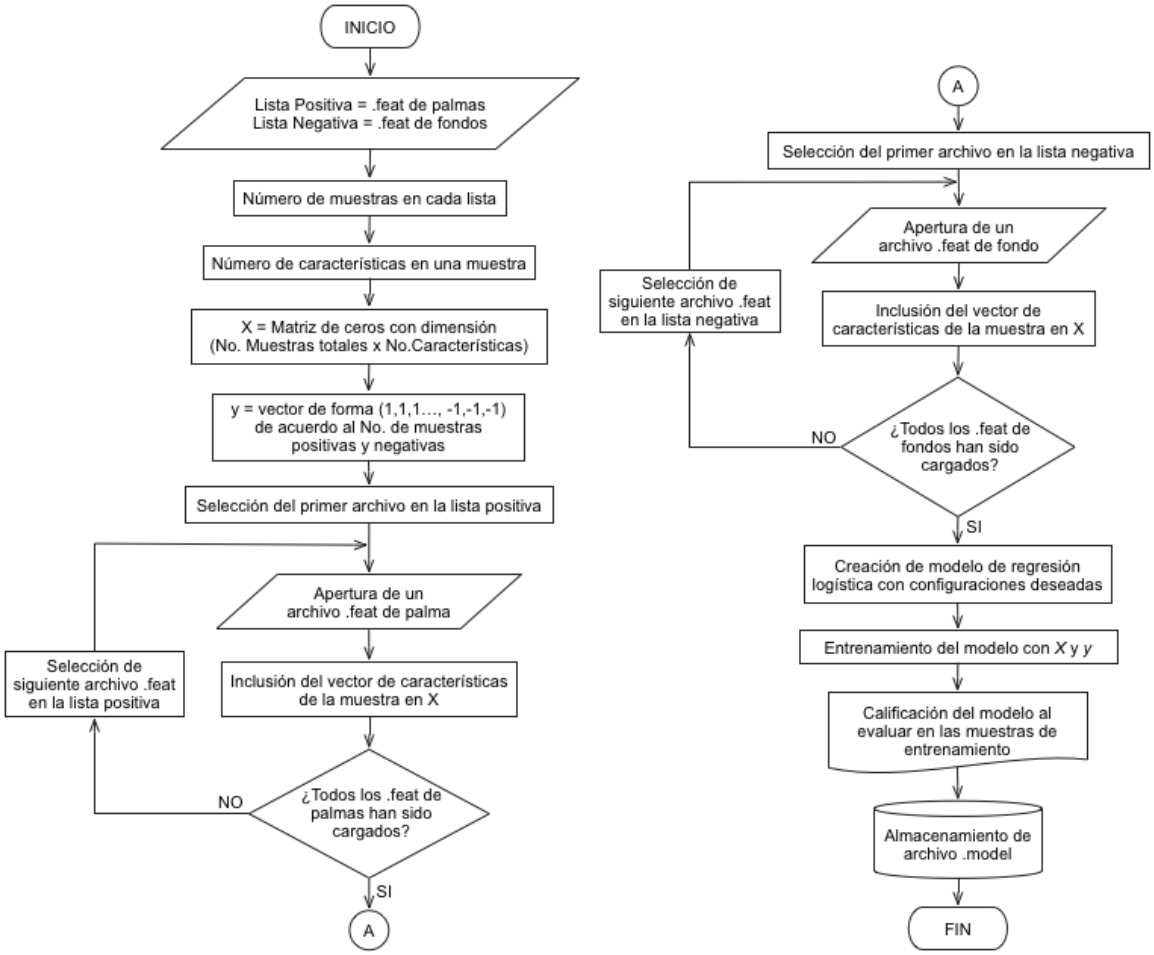


9.2.4 Entrenamiento de modelo de regresión logística

Una vez que las características de las muestras del conjunto de entrenamiento han sido descritas matemáticamente mediante el descriptor LBP, se utilizan los archivos ".feat" para cargar estas características en una matriz X que contenga la descripción de todas las muestras. Adicionalmente, de estos vectores se obtiene el número exacto de muestras positivas y negativas existentes, lo que permite construir el vector y con las etiquetas de clasificación correctas necesarias para el aprendizaje supervisado: 1 para las muestras positivas y -1 para las negativas.

La Figura 60 describe el proceso realizado para organizar estas características y entrenar, a partir de ellas, un modelo inicial de regresión logística, que obtiene un score de 1.0 al evaluar las muestras de entrenamiento, lo que indica, que es capaz de clasificar correctamente todas las muestras que usó para entrenarse.

Figura 60. Proceso de entrenamiento de modelo de regresión logística



El código de implementación de la librería scikit-learn para la creación del objeto que representa el modelo de regresión logística, perteneciente a la clase de modelos lineales, junto al método "*fit*" que permite entrenarlo de acuerdo a las muestras dadas, es presentado en la Figura 61.

Figura 61. Código de implementación de la librería scikit-learn para la creación y entrenamiento del modelo de regresión logística

```
# Train the classifier with X and y, and some given parameters
print 'Training linear model....'
model = linear_model.LogisticRegression(C=cfg.logReg_C,
                            penalty=cfg.logReg_penalty,
                            dual=cfg.logReg_dual,
                            tol=cfg.logReg_tol,
                            fit_intercept=cfg.logReg_fit_intercept,
                            intercept_scaling=cfg.logReg_intercept_scaling)

model.fit(X, y)

#Obtain the model score for the training set
score = model.score(X, y)
```

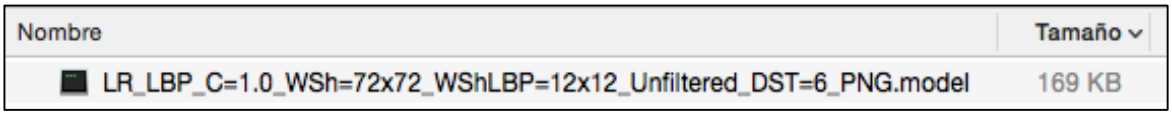
Todos los parámetros que configuran las distintas etapas del sistema se encuentran definidos en un archivo dedicado exclusivamente a esta tarea y son importados en cada programa que los requiere. En el caso del entrenamiento del modelo, la Figura 62 muestra los valores de los parámetros asignados que son usados en el código de la Figura 61.

Figura 62. Parámetros de entrenamiento del modelo definidos en el archivo de configuraciones

```
# Logistic Regression parameters
logReg_C = 1.0
logReg_penalty = 'l2'
logReg_dual = False
logReg_tol = 0.0001
logReg_fit_intercept = True
logReg_intercept_scaling = 100
```

Como resultado del proceso de entrenamiento se guarda un archivo con extensión ".model" que es cargado cada vez que se requiere que el modelo genere una decisión con respecto a una ventana. El nombre del archivo resalta algunas de las configuraciones asignadas para el sistema, como se observa en la Figura 63 .

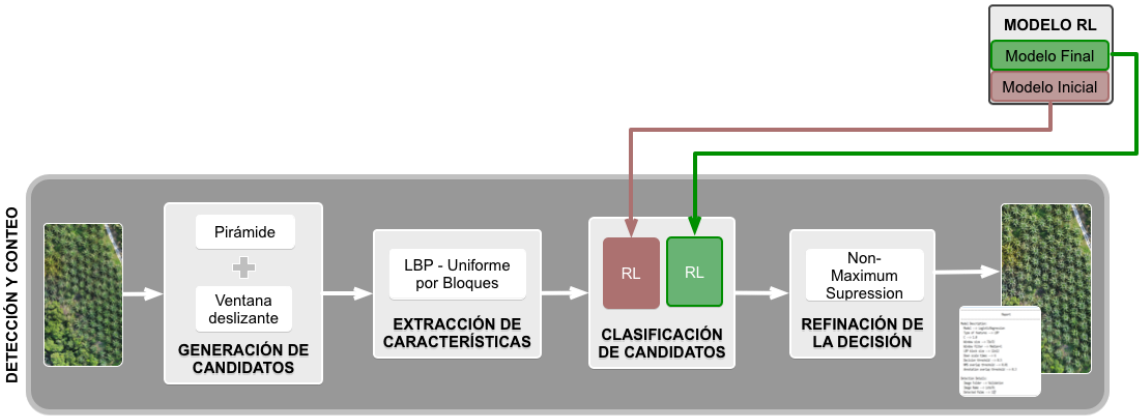
Figura 63. Visualización en explorador del archivo .model que contiene el modelo inicial



9.3 PROGRAMA DE DETECCIÓN Y CONTEO DE PALMAS EN UNA IMAGEN: MODELO PRELIMINAR

El diagrama de bloques de la Figura 64 presenta el orden de aplicación de las técnicas usadas para evaluar una o varias imágenes provenientes del software de fotogrametría. Cada ortomosaico debe pasar a través del proceso de forma individual.

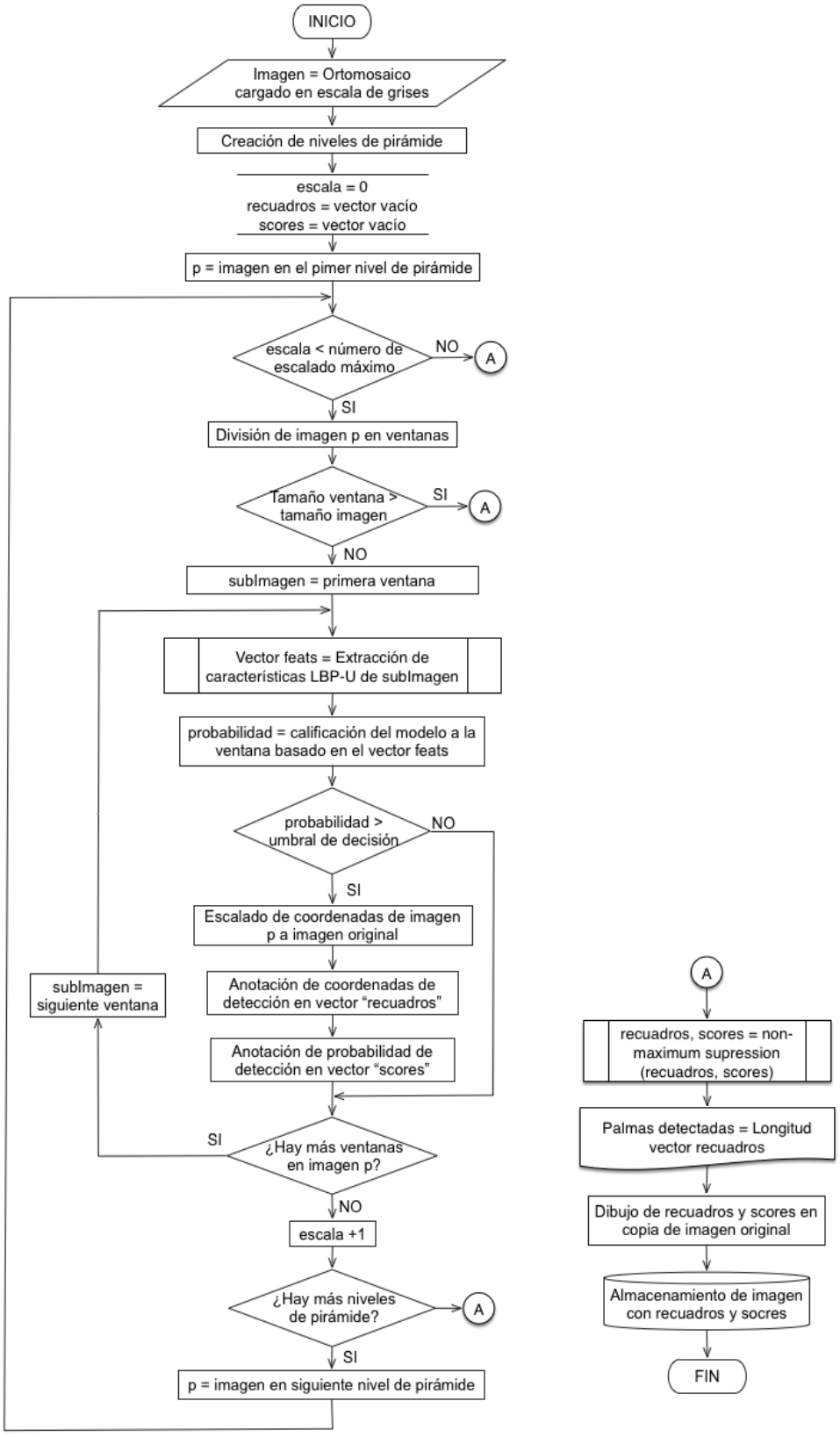
Figura 64. Diagrama de bloques del software de detección y conteo de palmas en una imagen



Para aplicar estas técnicas se desarrolla el algoritmo presentado en la Figura 65, en donde se genera una pirámide que contiene diferentes tamaños de la imagen original en escala de grises; cada uno de estos niveles es subdividido en ventanas con la misma dimensión que las muestras en la base de datos, a las cuales se les obtiene su vector de características LBP-U para que el clasificador entrenado previamente les asigne una probabilidad de contener palma.

Finalmente, todas las ventanas detectadas, junto a sus calificaciones, pasan a un algoritmo de non-maximum suppression que elimina las redundantes dejando sólo un recuadro de detección para cada palma encontrada. A partir de estas detecciones, se reporta el conteo de las palmas detectadas en la imagen y se dibujan los recuadros en una copia de la imagen original que se almacena en una carpeta de resultados.

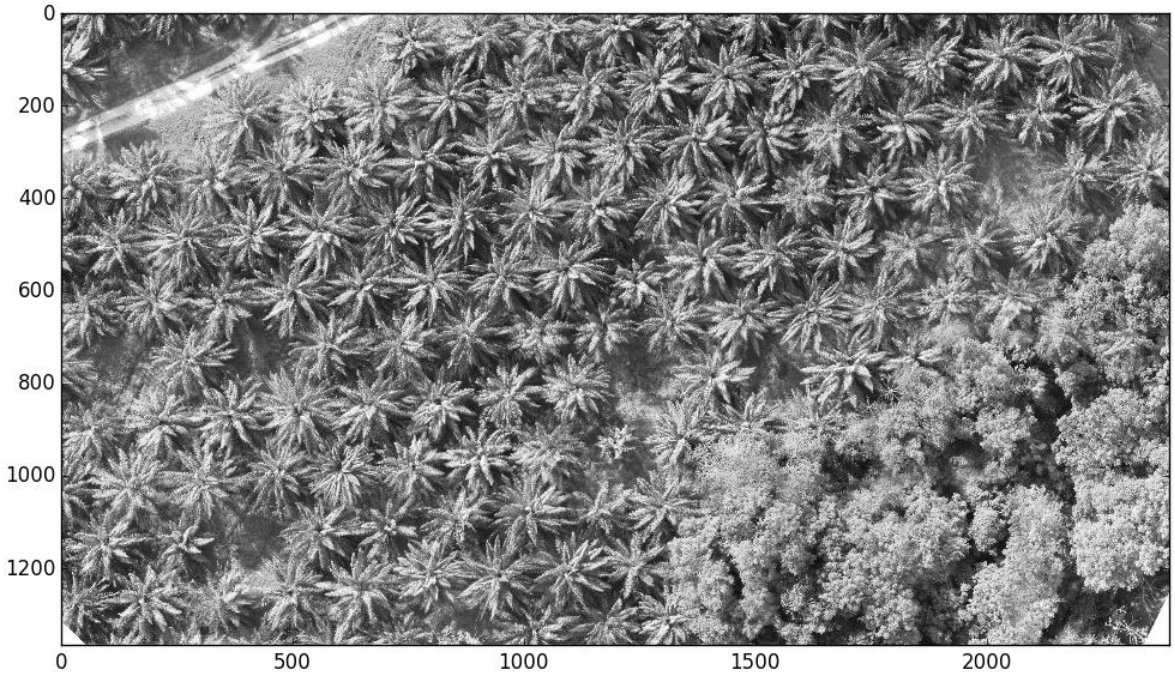
Figura 65. Algoritmo de detección de palmas en una imagen



9.3.1 Generación de candidatos

El ortomosaico proveniente de la herramienta de fotogrametría es cargado inicialmente como una matriz y convertido de RGB a escala de grises, como se muestra en la imagen de ejemplo en la Figura 66. El objetivo de la generación de candidatos es identificar todas las posibles ventanas en la imagen, con el tamaño establecido en el archivo de configuraciones, donde puede ubicarse una palma, es decir, que tengan características similares a las de las muestras positivas de entrenamiento. Para el modelo inicial planteado, esta ventana se ha definido con una dimensión de 72x72 pixeles.

Figura 66. Ortomosaico como imagen en escala de grises



Los ortomosaicos obtenidos con la herramienta de fotogrametría son ajustados para tener una escala definida (cm x pixel) y en la captura de imágenes el UAV siempre vuela a la misma altura. De esta forma, se asegura que el tamaño de un cuadro que encierra a una palma adulta vista desde el aire varía aproximadamente entre los 50 a 90 pixeles cuadrados, pero no alcanza valores muy por encima de este rango.

Sin embargo, algunas palmas son más grandes que otras o pueden identificarse fácilmente con ventanas un poco más grandes. Por esta razón, se genera un escalado de la imagen conocido como pirámide, que se estructura como lo

presenta la Figura 67. El nivel 0 corresponde a la imagen original, y cada nivel siguiente contiene la imagen suavizada por un filtro gaussiano y luego reducida en un factor de 1.1 para cada eje. La pirámide es implementada mediante la herramienta "*pyramid_gaussian*" de la librería scikit-image como se presenta en la Figura 68 , lo que permite obtener como resultado una tupla que guarda en cada posición la imagen de un nivel de la pirámide.

Figura 67. Estructura de la pirámide para generación de candidatos

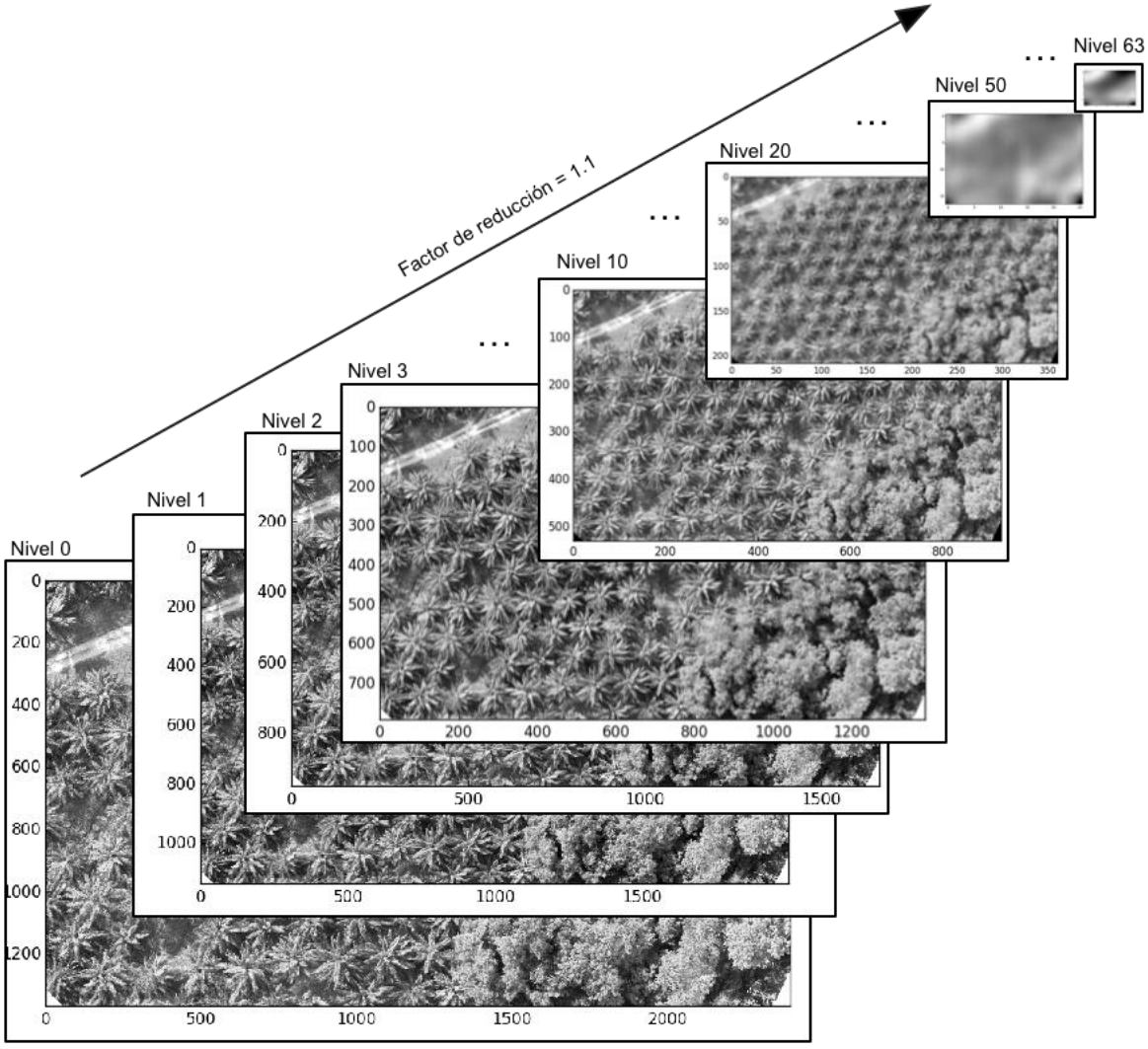


Figura 68. Código de implementación de la pirámide gaussiana

```
pyramid = tuple(pyramid_gaussian(image, downscale=cfg.downScaleFactor))
```

Los niveles más altos de la pirámide permiten identificar palmas más grandes, y debido a la limitación del tamaño de palmas que pueden presentarse, se establece un valor máximo de escalados de forma que sólo se recorrerá la pirámide hasta el nivel ajustado en el archivo de configuraciones, evitando cómputos innecesarios y evadiendo la posibilidad de detecciones falsas que causen confusión. En el modelo inicial, el DST ("*Down Scale Times*") es ajustado a un valor de 6 de forma que sólo se considera hasta el nivel 5 de la pirámide.

Cada nivel de la pirámide es recorrido con una ventana deslizante en búsqueda de candidatos. Esta técnica se logra con la herramienta "*view_as_windows*" de la librería scikit-image que subdivide la imagen en ventanas de tamaño definido (72x72 para el modelo inicial) con un paso de 2 veces el tamaño de los bloques LBP-U (es decir, 24 en el caso inicial de bloques LBP-U de 12x12), de esta forma cada ventana se desplaza de la anterior el equivalente a dos histogramas. La Figura 69 presenta la implementación de esta herramienta con la excepción de error que se genera en caso de que la ventana sea mayor que la imagen que se intenta dividir. Los valores asignados a los parámetros en el archivo de configuraciones se muestran en la Figura 70.

Figura 69. Implementación de la herramienta "*view_as_windows*" para la técnica de ventana deslizante

```
try:
    views = view_as_windows(p, cfg.window_shape, step=cfg.window_step)
except ValueError:
    #Block shape is bigger than image
    break
```

Figura 70. Configuraciones del tamaño y paso de la ventana deslizante

```
# Size of windows for the sliding window on the test images
window_shape = (72, 72)
window_margin = lbp_win_shape[0]
window_step = lbp_win_shape[0]*2
```

Como resultado de este proceso, en el momento en que cada imagen de la pirámide es analizada, se genera una matriz de dimensión $(m \times n)$, de acuerdo al número de ventanas que quepan en el nivel correspondiente, en donde cada posición contiene una ventana candidata a ser palma. Por ejemplo, en el caso de la imagen de la Figura 66, el nivel 0 de la pirámide, es decir, el que tiene el mismo tamaño que la ventana original, genera una matriz con una dimensión de 55x98x72x72, lo que se traduce en 5390 ventanas de 72x72 que son evaluadas para ese nivel de la pirámide.

9.3.2 Extracción de características

Cada ventana candidata debe pasar por el mismo proceso de extracción de características LBP-U por el que pasan las ventanas de entrenamiento, descrito en la sección 9.2.3 Extracción de características en la página 82. De esta forma, se genera un vector de características de 7139 valores (121 bloques LBP x 59 características cada uno) que describe su textura y le permite al modelo entender el tipo de objeto que contiene.

9.3.3 Clasificación de candidatos

El objeto que representa al modelo entrenado es cargado al software de detección para que genere una calificación a cada ventana analizada. Para esto, se utiliza el método "*decision_function*" que usa el vector de características de la ventana para generar un puntaje o score que indica la probabilidad de que la ventana analizada contenga una palma. Matemáticamente, corresponde a la distancia con signo que existe entre esa muestra y el hiperplano que actúa como frontera multidimensional entre las dos categorías. Por tanto, las calificaciones positivas son distancias hacia las muestras positivas o palmas, mientras que las negativas se inclinan hacia la categoría de fondo. Mientras más alto sea este valor, más "seguro" se encuentra el modelo de la existencia de una palma en esa ventana. La Figura 71 muestra la implementación de este método para obtener el puntaje de calificación en una ventana.

Figura 71. Implementación del método "*decision_function*" para la obtención del puntaje dado por el modelo para una ventana

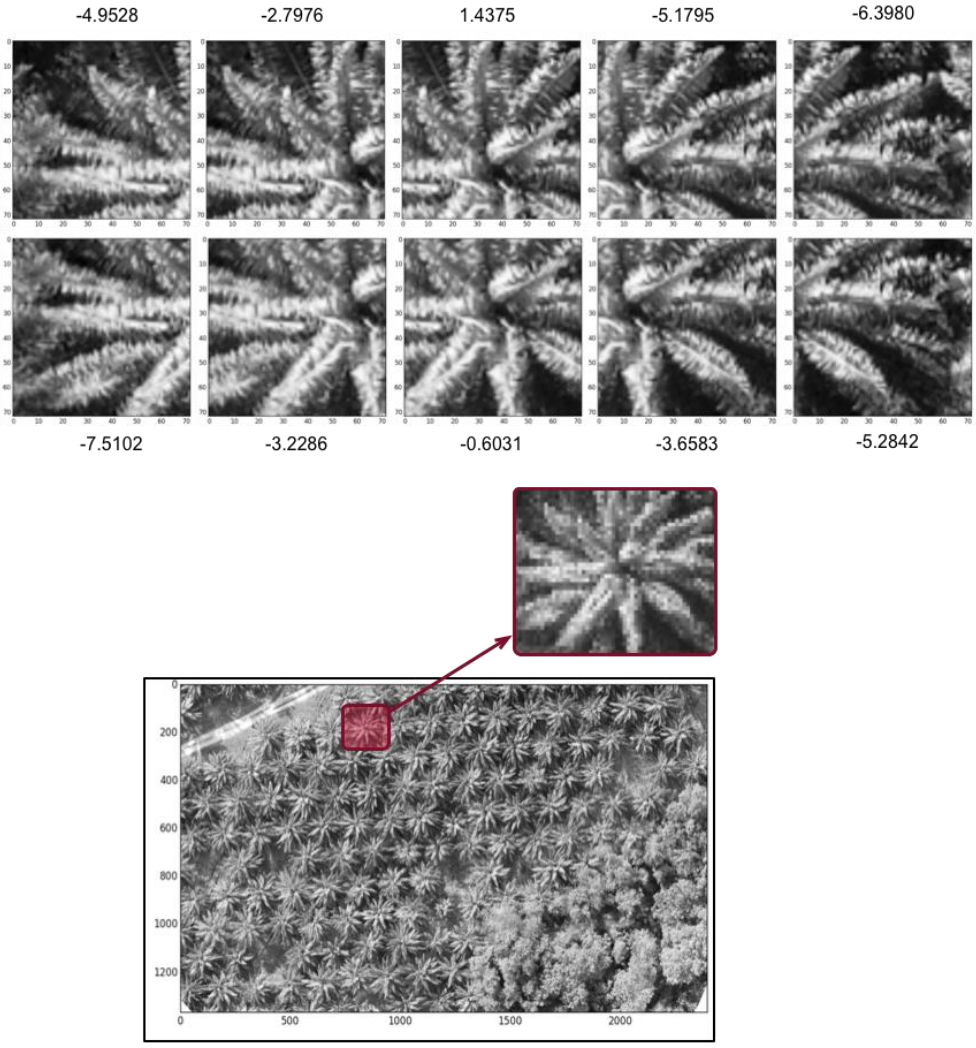
```
#Obtain prediction score
decision_func = model.decision_function(feats)
```

La Figura 72 presenta el puntaje dado por el modelo para algunas ventanas de ejemplo, cuando se desplaza por la zona indicada en el rectángulo rojo. Se observa que los dos mejores puntajes han sido obtenidos por las ventanas que contienen el núcleo de la palma más centrado en el cuadrado; sin embargo sólo una de ellas ha obtenido una calificación positiva.

Cuando el puntaje asignado por el modelo supera el umbral de decisión definido como punto de operación en el archivo de configuraciones, se considera que la ventana califica dentro de la categoría palma y sus coordenadas son almacenadas en el vector "recuadros", mientras que su puntaje es respectivamente guardado en el vector "scores". De esta forma, al finalizar las ventanas deslizantes de los

distintos niveles de la pirámide, se obtienen las coordenadas de las ventanas que han sido consideradas como palmas y el puntaje de cada una.

Figura 72. Puntajes dados por el modelo inicial para ventanas deslizantes en un área definida



9.3.4 Refinación de la decisión

Debido a las técnicas de ventana deslizante y pirámide utilizadas para la generación de candidatos, es muy probable que una misma palma sea detectada por varios recuadros que obtuvieron la calificación suficiente para superar el umbral. Por eso es necesario refinar la decisión mediante la técnica de non-maximum suppression.

El algoritmo recibe los vectores de “recuadros” y “scores” que contienen, respectivamente, las coordenadas y los puntajes de las detecciones que superan el umbral. Ambos son organizados de acuerdo al orden descendente de los puntajes, de forma que los primeros recuadros en analizarse son aquellos con mayor puntaje.

Una nueva lista de recuadros y puntajes es creada en donde se almacenan las detecciones finales. Cada recuadro es revisado para calcular su solapamiento con los bloques que ya han pasado a la nueva lista. Si no existe ningún recuadro con quien se solape en una proporción mayor a la establecida en el archivo de configuraciones, el recuadro analizado pasa a ser parte de la lista de recuadros finales de detección. Pero, si por el contrario, en la lista final ya existe un recuadro con el cual se solapa, la detección que está siendo analizada es desechada. La organización previa de acuerdo al puntaje asegura que los primeros recuadros en entrar a la lista final siempre son aquellos con el mayor puntaje para cada palma, y que los que se solapan con ellos son detecciones con menor calificación que pueden ser eliminadas.

El porcentaje de solapamiento tolerado por el NMS es ajustado a 1% en el archivo de configuraciones, lo que significa que los recuadros cuyas áreas se solapen en más del 1% se considera que pertenecen a la misma palma, y sólo debe dejarse aquel con mayor puntaje. Este umbral es bastante bajo ya que las palmas nunca se encuentran tan cercanas las unas a las otras, y el tamaño de las ventanas cubre el centro de la palma y pocas ramificaciones, de forma que los verdaderos recuadros con palma no deben solaparse. Esta medida ayuda a eliminar los errores que pueden presentarse cuando el modelo selecciona como palma algunos de los “fondos difíciles” como las ventanas que incluyen las ramas entre palmas, ya que la textura de estas puede llegar a confundirse con las palmas.

La Figura 73 presenta los recuadros obtenidos en la imagen antes de aplicar el NMS, en la cual se detectan un total de 410 objetos, que se reducen a 112 después de refinar la decisión, generando la imagen que se presenta en la Figura 74.

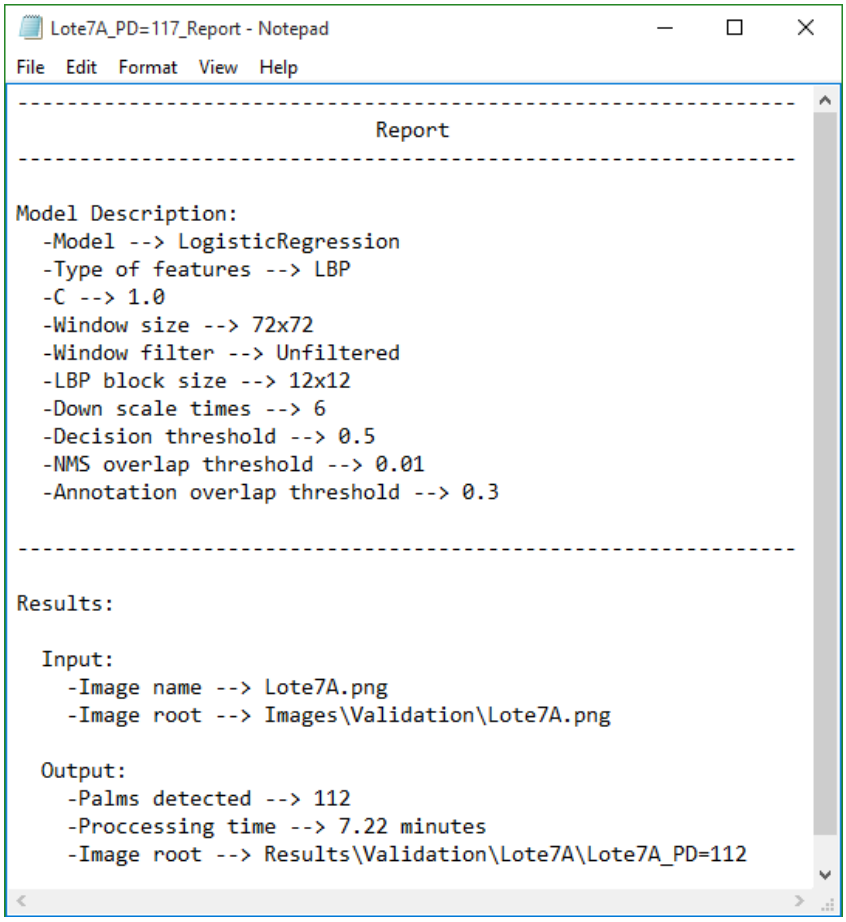
Figura 73. Imagen con detecciones obtenidas antes de aplicar non-maximum suppression

Figura 74. Imagen con detecciones obtenidas después de aplicar non-maximum suppression

9.3.5 Presentación de resultados de la detección

Una vez que se ha refinado la decisión y se han decidido los recuadros finales que contienen palmas, el largo de este vector indica el número de palmas detectadas para la imagen analizada. Este resultado se presenta en un reporte almacenado en un archivo de texto como se presenta en la Figura 75.

Figura 75. Reporte de detección realiza por el modelo preliminar en imagen de ejemplo



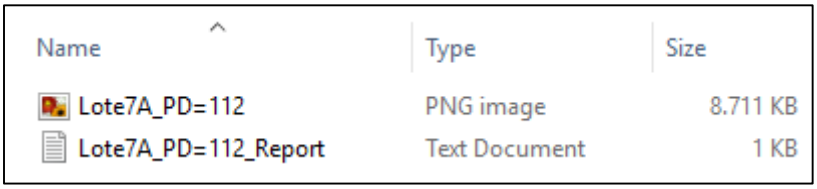
Adicionalmente, se crea un objeto de dibujo perteneciente a la clase "*Draw*" de la librería PIL (*Python Imaging Library*) que permite dibujar líneas en la imagen al darle las coordenadas de inicio y finalización de las mismas. De esta forma se utilizan las coordenadas en el vector de "recuadros" para dibujar las líneas que enmarcan cada una de las detecciones.

La librería PIL también permite agregar texto a la imagen en una coordenada específica usando la función "*truetype*" incluida en el módulo de *ImageFont*. Esta

característica es aprovechada para escribir en cada recuadro de detección dibujado, el puntaje dado por el modelo.

La imagen obtenida como resultado luce como la presentada en la Figura 74 y es almacenada en un archivo con extensión ".png" cuyo nombre incluye el número de palmas encontradas, así como lo presenta la Figura 76 .

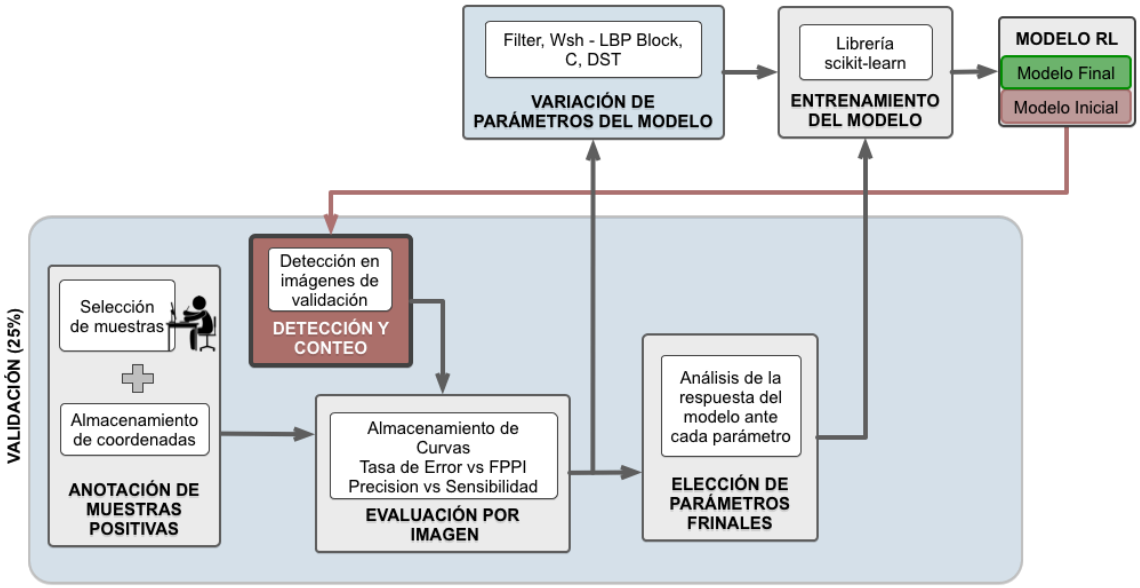
Figura 76. Vista desde el explorador de la imagen resultado de la detección con el modelo preliminar



9.4 VALIDACIÓN DEL SISTEMA

El proceso de entrenamiento permite obtener un modelo inicial con la capacidad de clasificar ventanas en las dos categorías deseadas. La fase de validación busca evaluar y optimizar el rendimiento del modelo en nuevas imágenes. La Figura 77 describe la secuencia de tareas que componen este proceso.

Figura 77. Diagrama de bloques del proceso de validación



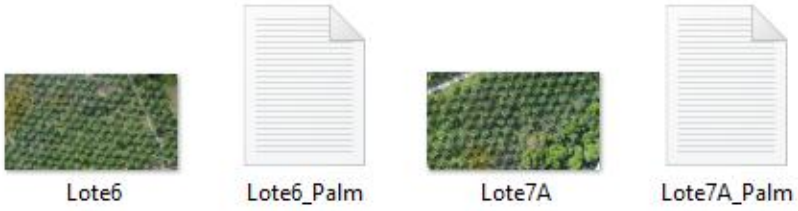
9.4.1 Anotación de muestras positivas

Para evaluar las clasificaciones realizadas por el programa de detección en una imagen, es necesario tener una referencia de las ventanas que corresponden a las detecciones correctas para poder compararlas. Por esta razón, una persona debe seleccionar en los ortomosaicos del conjunto de validación, la ubicación de cada una de las palmas para que estas sean almacenadas en un archivo y puedan ser revisadas posteriormente.

Este proceso utiliza el mismo programa que la anotación realizada para las imágenes de entrenamiento, el cual es descrito en la sección 9.2.1 Anotación de muestras positivas y negativas en la página 75. La única diferencia es que en el proceso de validación solo es necesario anotar las muestras positivas, es decir, las ubicaciones de las palmas, ya que estos son los cuadros que se espera que el programa obtenga y con los cuales se realiza la evaluación.

Como resultado, se obtiene el archivo de texto con las anotaciones de las muestras positivas para cada ortomosaico almacenado en la misma carpeta que la imagen, como lo presenta la Figura 78.

Figura 78. Archivos de texto con las anotaciones positivas de las imágenes de validación



9.4.2 Evaluación por imagen

La evaluación por imagen busca generar una representación del desempeño del modelo entrenado cuando clasifica nuevas imágenes, a través de la creación de dos gráficas que muestran la relación entre diversas métricas al variar el punto de operación del modelo, es decir, con diferentes umbrales de decisión.

Se inicia con la aplicación del programa de detección de palmas (planteado en la sección 9.3) en cada uno de los ortomosaicos que pertenecen al conjunto de evaluación, pero sin la ejecución del algoritmo de non-maximum suppression para obtener todas las detecciones posibles y con un umbral de decisión de -2 para que

pasen todas las palmas y varios ruidos, desechando sólo aquellas ventanas que tienen una probabilidad muy alta de ser fondo.

Los vectores de “recuadros” y “scores” obtenidos, junto al directorio y nombre de cada ortomosaico, son almacenados en un diccionario que se guarda en un archivo con formato “.result”.

A partir de estos resultados, se realiza para cada imagen, una variación del umbral de decisión desde -2 hasta 2 con pasos de 0.02. Esto genera varios puntos de operación del modelo y en cada uno, las ventanas que superan el umbral pasan a través del algoritmo de non-maximum suppression eliminando la redundancia y dejando sólo una ventana por cada ubicación donde considera que se encuentra una palma.

Estas ubicaciones son comparadas con las coordenadas anotadas por la persona para cada imagen calculando el área de solapamiento de cada una de ellas: cuando el solapamiento es mayor al 0.3 (30% del área anotada) se considera que la detección de la ventana es correcta. Estas comparaciones permiten establecer los verdaderos positivos (TP, *true positives,* ventanas consideradas palmas que sí lo son), los falsos negativos (FN, *false negatives,* ventanas con palmas que no fueron detectadas) y falsos positivos (FP, *false positives*, ventanas detectadas como palmas pero que no lo son) para cada punto de operación.

Estos parámetros permiten calcular los FPPI (Falsos Positivos Por Imagen), la tasa de error, la tasa de detección o recall, y la precisión para cada umbral. La representación gráfica de estos parámetros genera las curvas *FPPI vs Miss Rate* (Tasa de Error) y *Recall vs Precision*, que determinan la evaluación por imagen para el modelo describiendo el comportamiento del mismo ante nuevas imágenes y dando la posibilidad de establecer el mejor punto de operación, el cual corresponde al que mas se acerque al punto ideal en cada gráfica: (0,0) en el caso de *FPPI vs Miss Rate* y (1,1) para *Recall vs Precision*.

9.4.3 Variación de parámetros

Para optimizar el rendimiento del modelo ante nuevas muestras, se varían algunos parámetros que determinan varias etapas del sistema definiendo el efecto que tiene cada uno de ellos en la evaluación y eligiendo el más adecuado.

Cada vez que se ajusta un parámetro, es necesario realizar de nuevo las operaciones que lo involucran y entrenar un nuevo modelo. La evaluación de cada

modelo es almacenada en un archivo de texto, de forma que puede ser graficada posteriormente y comparada con los modelos resultantes al variar el parámetro en estudio.

Los parámetros analizados y los respectivos valores asignados en el modelo inicial son:

- Filtro aplicado a la ventana = Sin filtro
- Tamaño de la ventana y del bloque LBP = 72x72 con bloque LBP 12x12
- Parámetro de regularización del modelo (C) = 1.0
- Número de escalado de pirámide (DST) = 6

Los parámetros son analizados en el orden planteado: se realizan las variaciones para cada uno, se entrenan los respectivos modelos, se evalúan y se comparan para elegir el mejor valor. A medida que el ajuste óptimo es elegido en cada parámetro, este se mantiene para el análisis del siguiente ya que están organizados en orden descendente de acuerdo a la incidencia que tiene cada uno sobre los demás.

1. FILTRO APLICADO A LA VENTANA
   Inicialmente, a la ventana de 72x72 usada para entrenamiento y para generación de candidatos, se le extraen las características LBP sin pasar por ningún tipo de procesamiento previo.

   Sin embargo, el uso de filtros que suavicen la imagen pueden hacer la ventana menos susceptible a la detección de ruido, por esto se plantea la aplicación de un filtro antes de la extracción de características. Este proceso debe realizarse tanto para las ventanas candidatas en el programa de detección, como para las ventanas de entrenamiento, lo que hace que se deba reentrenar el modelo para cada filtro.

   Las opciones planteadas son:
   - Filtro gaussiano con desviación estándar σ = 0.4
   - Filtro gaussiano con desviación estándar σ = 1
   - Filtro de mediana con máscara en forma de disco de radio= 1
   - Filtro de mediana con máscara en forma de disco de radio = 2
   - Sin filtro (ajuste inicial)

   La Figura 79 presenta los efectos de los filtros aplicados en los parámetros de FPPI y tasa de error, en donde las ventanas con filtro gaussiano más suave y sin filtro generaron el peor rendimiento, mientras que la aplicación

del filtro de mediana cuya máscara en forma de disco tiene radio = 1 presenta el mejor comportamiento y ofrece el punto de operación más cercano al punto ideal. Los efectos en la precisión y en la tasa de detección mostrados por la Figura 80 confirman el análisis. Por tanto, se incluye la aplicación del filtro de mediana con máscara en forma de disco de radio = 1 en el archivo de configuraciones.

Figura 79. Efectos del filtro de ventana en el desempeño del modelo: *FPPI vs Miss Rate*

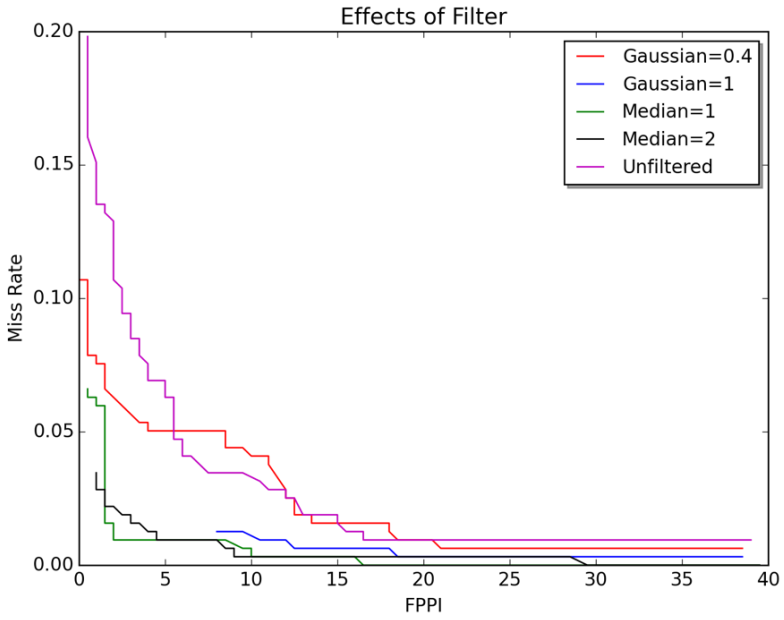
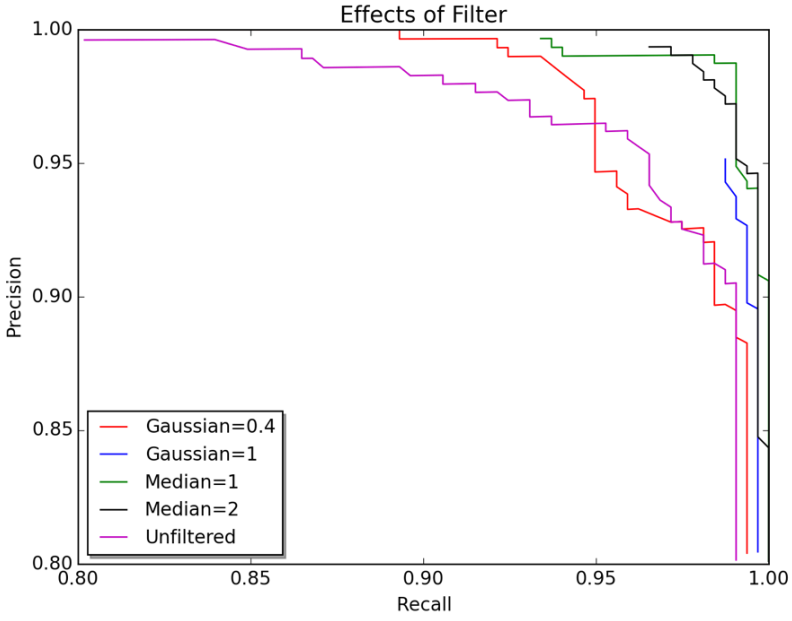


Figura 80. Efectos del filtro de ventana en el desempeño del modelo*: Recall vs Precision*



2. TAMAÑO DE LA VENTANA Y DEL BLOQUE LBP

En el modelo inicial la ventana que define una palma es ajustada a un tamaño de 72x72 pixeles que abarca el área de anotación de una palma; sin embargo, existen otros valores cercanos que enmarcan una zona similar y que deben ser considerados para definir el tamaño óptimo.

La ventana es subdividida en bloques LBP y por tanto su modificación no puede ser independiente a este parámetro. Juntos, conforman un factor decisivo en el sistema, ya que determinan los tamaños de las muestras de entrenamiento y la longitud de los vectores de características que definen a una ventana, teniendo una influencia directa en la cantidad de cómputo realizado, el tiempo empleado y la dimensión de la matriz con la que se entrena el modelo.

La variación de estos parámetros no sólo implica el reentrenamiento del modelo, sino también la extracción de las ventanas que conforman la base de datos para entrenamiento ya que el tamaño de la zona recortada es modificado.

Las opciones planteadas son:

- Ventana de 72x72 con bloques LBP 12x12 (ajuste inicial)
- Ventana de 72x72 con bloques LBP 8x8
- Ventana de 64x64 con bloques LBP 8x8
- Ventana de 60x60 con bloques LBP 12x12

La Figura 81 presenta los efectos de las dimensiones de la ventana y del bloque LBP en los parámetros de FPPI y tasa de error, de forma que la ventana de 72x72 con bloques LBP 12x12 presenta los puntos de operación más cercanos al punto ideal. El comportamiento del modelo con ventana de 60x60 y bloques LBP de 12x12 demuestra que el tamaño de los bloques LBP por sí solos no determina el modelo más apropiado. Los modelos con las ventanas de 72x72 presentaron los mejores comportamientos, sugiriendo que las ventanas grandes funcionan mejor que las pequeñas. Los efectos en la precisión y en la tasa de detección mostrados por la Figura 82 confirman el análisis. Por tanto, se mantiene el tamaño de la ventana y del bloque LBP planteado en el modelo inicial.

Figura 81. Efectos de la dimensión de la ventana y del bloque LBP en el desempeño del modelo: *FPPI vs Miss Rate*

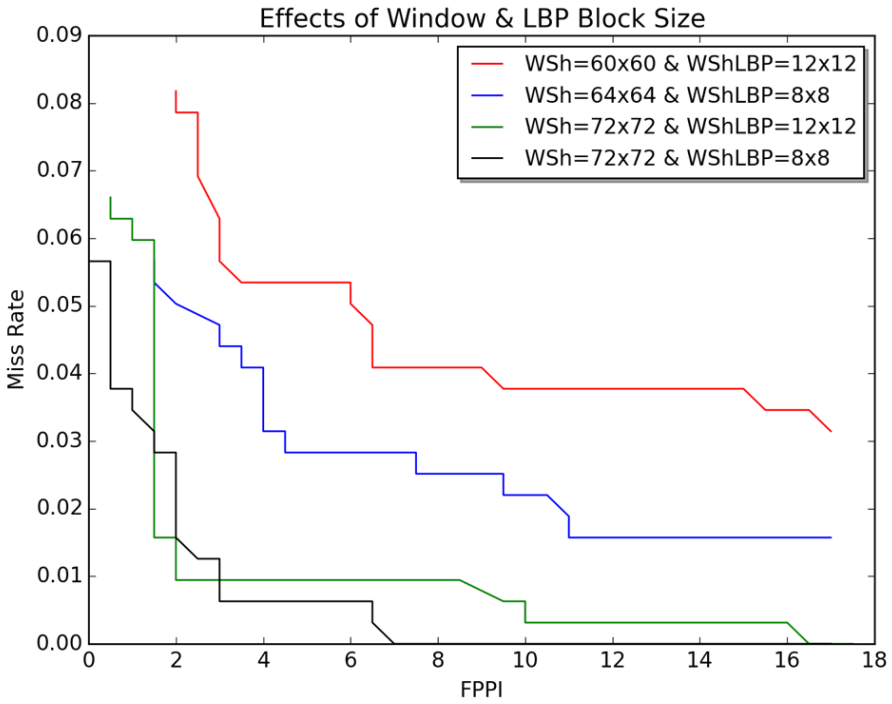
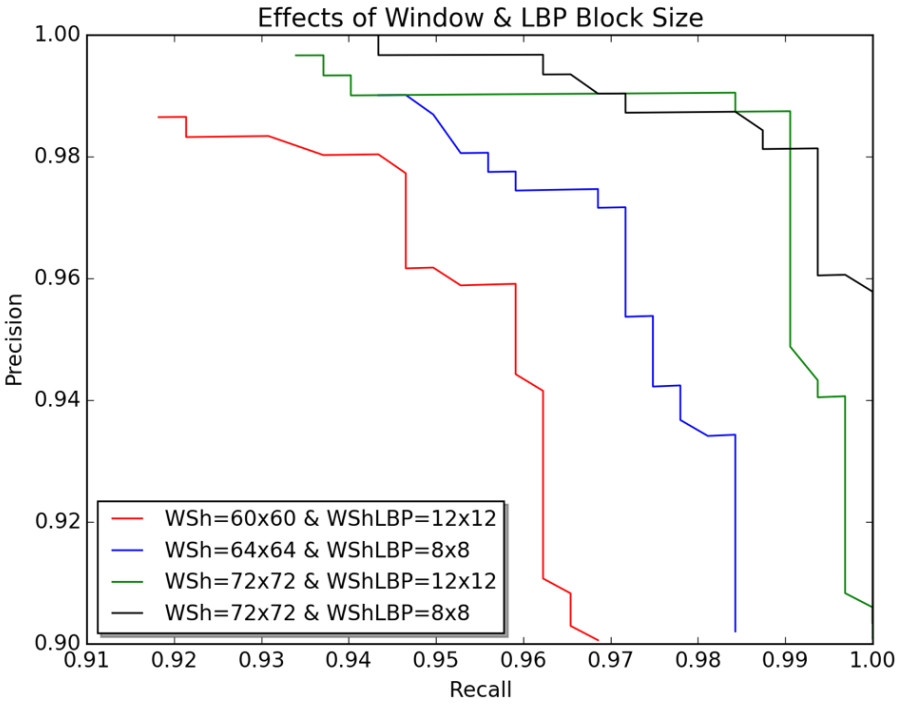


Figura 82. Efectos de la dimensión de la ventana y del bloque LBP en el desempeño del modelo: *Recall vs Precision*



3. PARÁMETRO DE REGULARIZACIÓN DEL MODELO (C)

   La regularización del modelo evita que el hiperplano que actúa como frontera entre las clases, el cual es desarrollado durante el entrenamiento, se sobreajuste a estas muestras disminuyendo la capacidad de generalización del modelo.

   En la definición del objeto de modelo de regresión logística durante el entrenamiento usando la librería scikit-learn, se define el parámetro C. Este corresponde al inverso de la fuerza de regularización. De esta forma, valores más pequeños de C especifican una regularización más agresiva.

   Las opciones planteadas son:

   - C = 0.01
   - C = 1.0 (ajuste inicial)
   - C = 100

   La Figura 83 presenta los efectos de la regularización del modelo (C) en los parámetros de FPPI y tasa de error, en donde se observa que el modelo regularizado con un parámetro de C=0.01 exhibe el peor comportamiento sugiriendo que la regularización es demasiado agresiva y causa que la frontera no se ajuste adecuadamente a los datos de entrenamiento de forma que no queda bien ubicada en el hiperplano y las predicciones no tienen la misma calidad que las de los otros parámetros.

   Por otra parte, los modelos con C=1.0 y C=100 se comportan de forma similar, aunque C=1.0 ofrece los puntos de operación más cercanos al punto ideal. Los efectos en la precisión y en la tasa de detección mostrados por la Figura 84 confirman el análisis. Por tanto, se mantiene el parámetro de regularización establecido en el modelo inicial C = 1.0, representando un nivel de regularización intermedio.

Figura 83. Efectos del parámetro de regularización C en el desempeño del modelo: *FPPI vs Miss Rate*

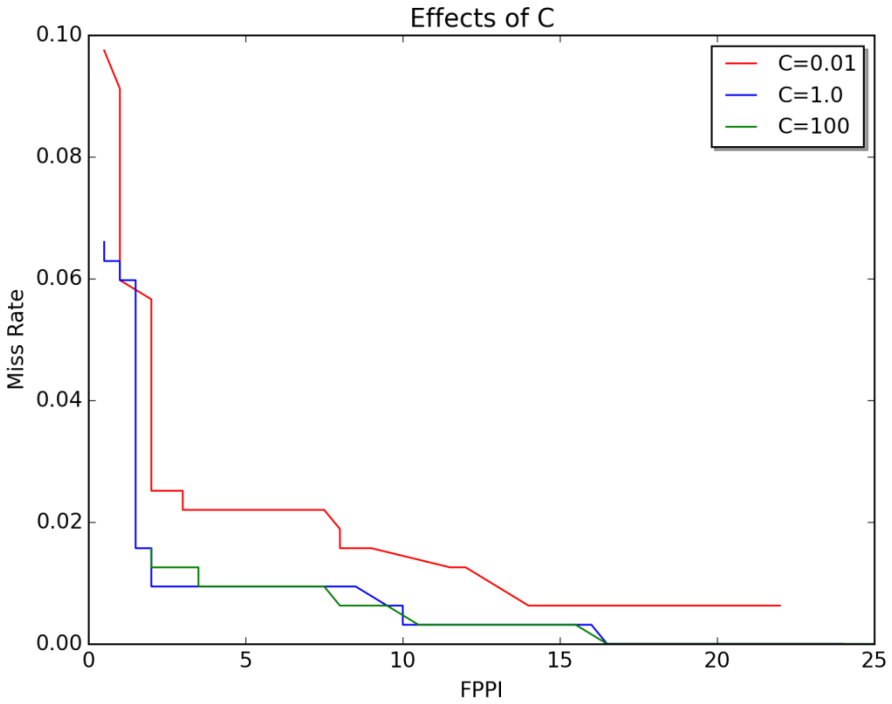
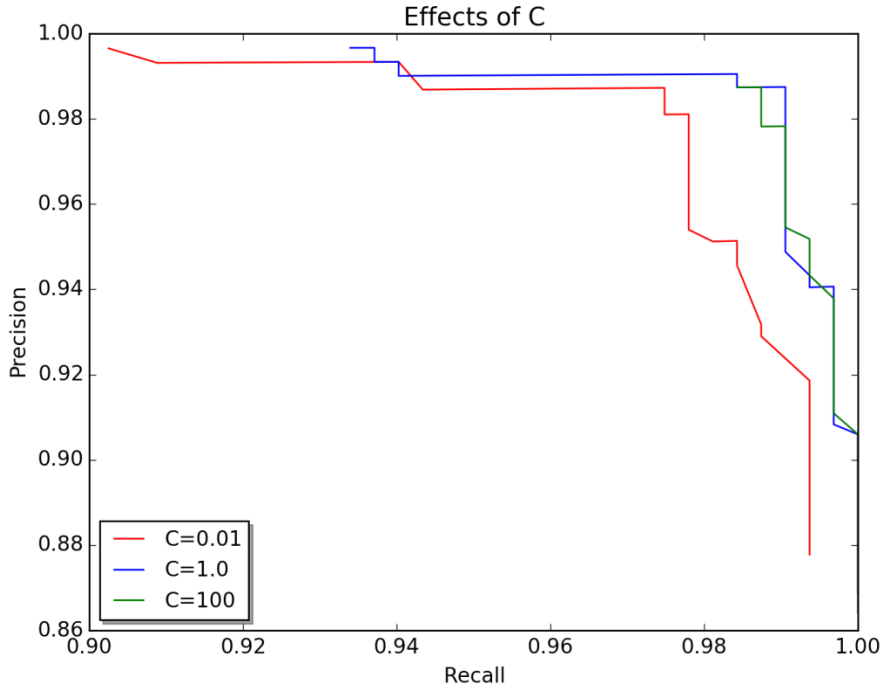


Figura 84. Efectos del parámetro de regularización C en el desempeño del modelo: *Recall vs Precision*



4. *DOWN SCALE TIMES* (DST)

   El número de veces que se escala la pirámide o *Down Scale Times* (DST) determina hasta que nivel de la pirámide se considera para aplicar ventana deslizante y generar candidatos para palmas.

   La importancia del DST en el caso de este sistema no radica principalmente en la detección de palmas mucho más grandes que otras ya que todas las palmas manejan un tamaño similar, sino en ampliar la posibilidad de ventanas candidatas cuando las obtenidas en la imagen de tamaño original no encajan exactamente en el centro de la palma. Por tanto, el rango de variación de los parámetros se mantiene dentro los primeros niveles ya que no se requiere llegar a niveles superiores que pueden ser contraproducentes al agregar más ruido a la decisión.

   Las opciones planteadas son:
   - DST = 4
   - DST = 6 (ajuste inicial)
   - DST = 8
   - DST = 10
   - DST = 12

   La Figura 85 presenta los efectos del DST en los parámetros de FPPI y tasa de error, en donde se observa que el modelo con DST=4 muestra el peor desempeño demostrando que el número de niveles considerados no fue suficiente. Los modelos con DST = 8, 10 y 12 exhiben un comportamiento muy similar sugiriendo que llega un punto en el que utilizar más niveles de la pirámide no aporta mejoras significativas. Estos modelos presentan los puntos de operación más cercanos al punto ideal. Sin embargo, la ejecución de los programas de detección para cada parámetro DST tuvo una duración aproximada de 5 a 6 minutos para DST=6, mientras que a partir del DST=8 el tiempo de análisis alcanza los 12 minutos. Los efectos en la precisión y en la tasa de detección mostrados por la Figura 86 confirman este análisis.

   Por tanto, se determina que las mejoras obtenidas por los modelos con DST>=8 no compensan el considerable aumento en el tiempo de procesamiento por cada imagen, haciendo que el sistema sea menos eficiente. Como consecuencia, se mantiene el DST=6 definido inicialmente.

Figura 85. Efectos del DST en el desempeño del modelo: *FPPI vs Miss Rate*

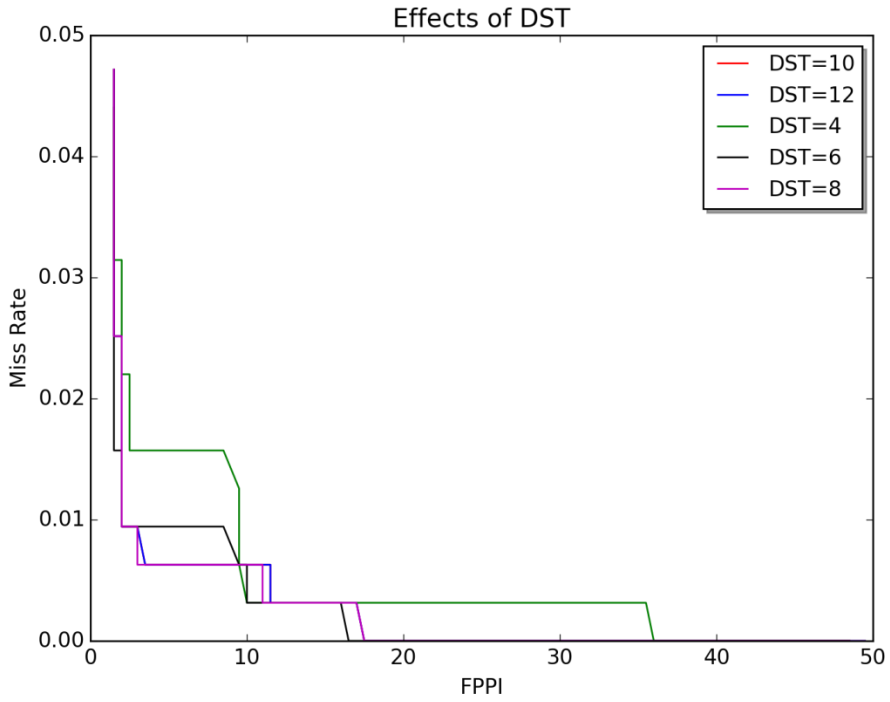
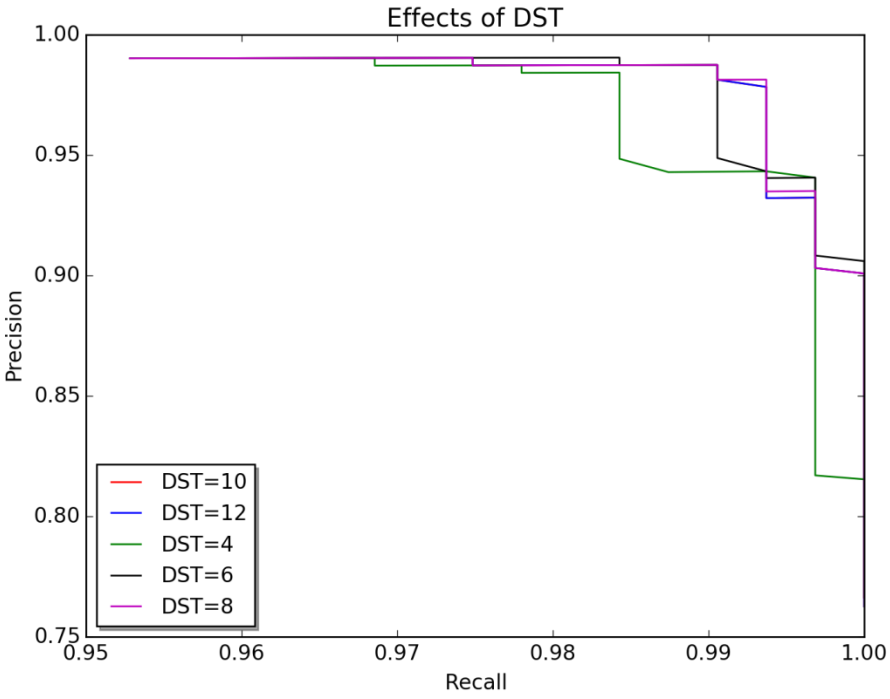


Figura 86. Efectos del DST en el desempeño del modelo: *Recall vs Precision*



9.4.4 Modelo final

Al comparar y analizar los efectos de los parámetros en el desempeño del sistema, se determina un modelo final con las siguientes configuraciones:

- Filtro aplicado a la ventana: Filtro de mediana con máscara en forma de disco de radio = 1
- Tamaño de la ventana y del bloque LBP: 72x72 con bloque LBP 12x12
- Parámetro de regularización del modelo: C = 1.0
- Número de escalado de pirámide: DST = 6

El principal cambio en los ajustes del modelo es la inclusión de un filtro de mediana con máscara en forma de disco de radio = 1 antes de la extracción de las características LBP a las ventanas en el entrenamiento y en la generación de candidatos.

La Figura 87 muestra una ventana de palma y una de fondo a las que se les ha aplicado el filtro elegido. Las imágenes LBP-U de las mismas son presentadas en la Figura 88, junto a los histogramas respectivos en la Figura 89. Al comparar estos histogramas con los presentados en la Figura 56, que corresponden a los de una ventana LBP-U de una palma y de un fondo sin filtro, se observa que la aplicación del mismo permite que la diferencia en la distribución del histograma de las dos clases sea más marcada, haciendo que sea más fácil para el modelo generar una frontera entre los dos y posteriormente clasificarlos.

Figura 87. Ventanas de Palma y Fondo después de aplicado el filtro seleccionado

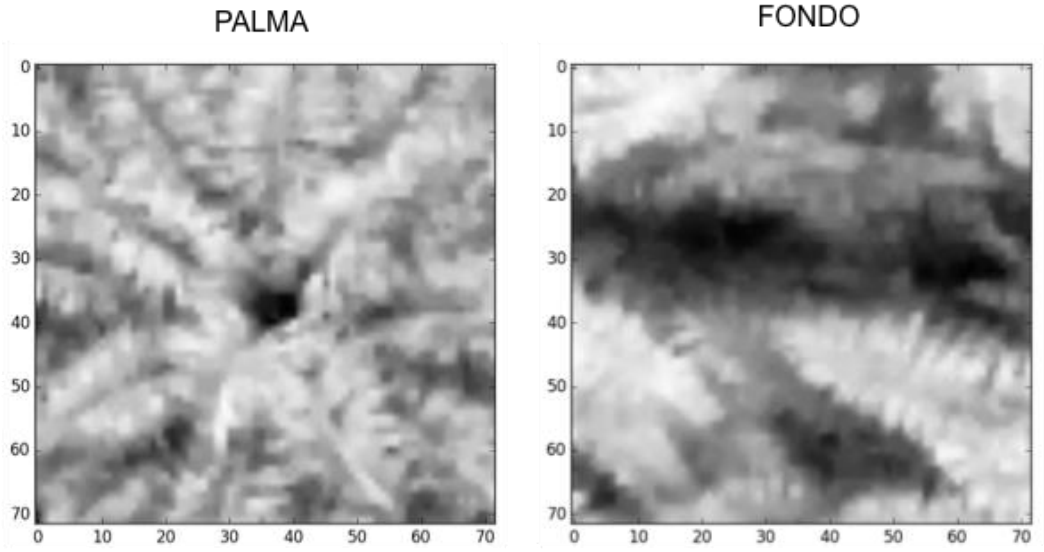


Figura 88. Imágenes LBP-U de las ventanas de Palma y Fondo obtenidas después de aplicar el filtro

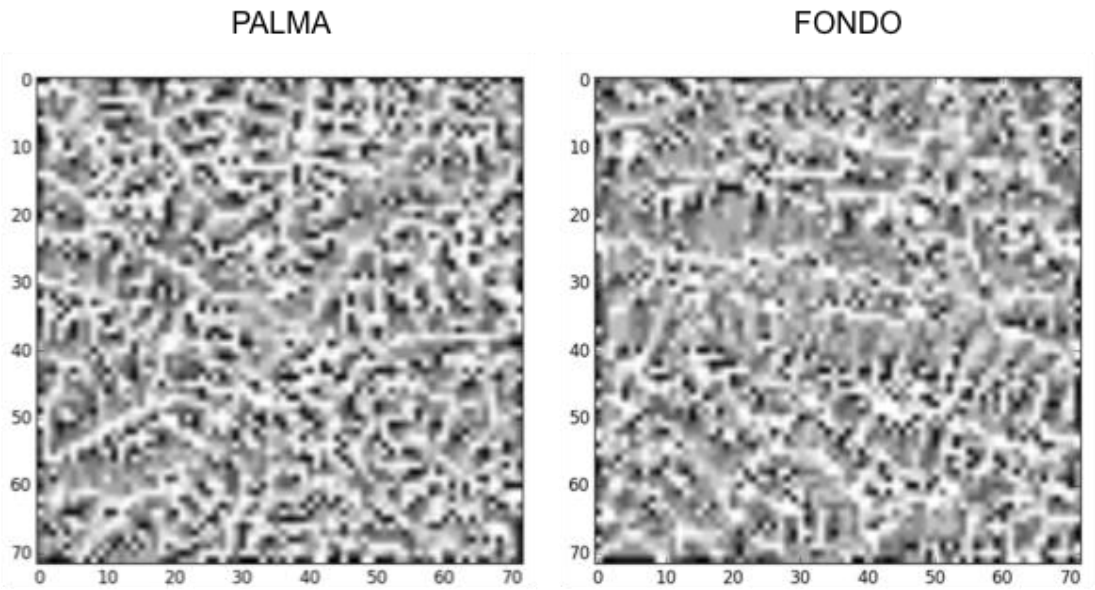
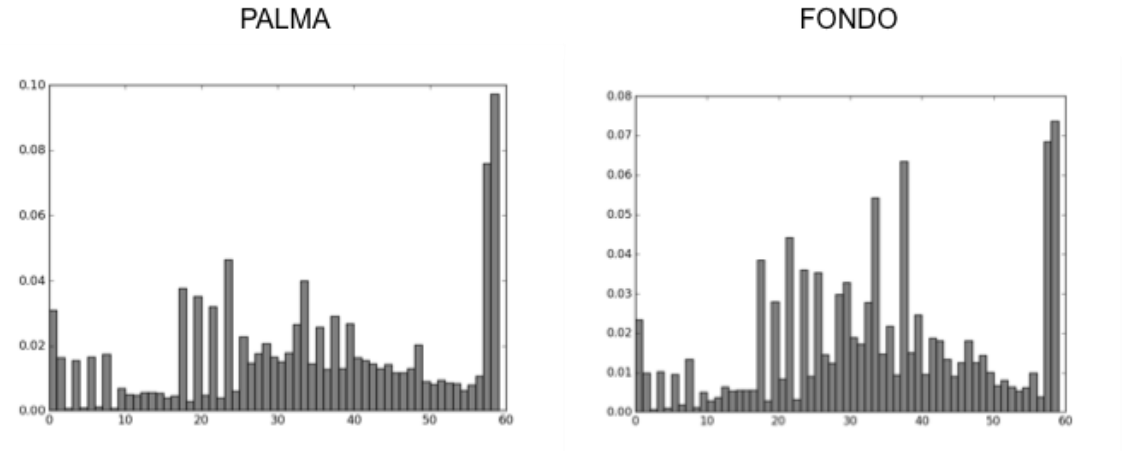


Figura 89. Histogramas de las imágenes LBP-U de ventanas con filtro aplicado

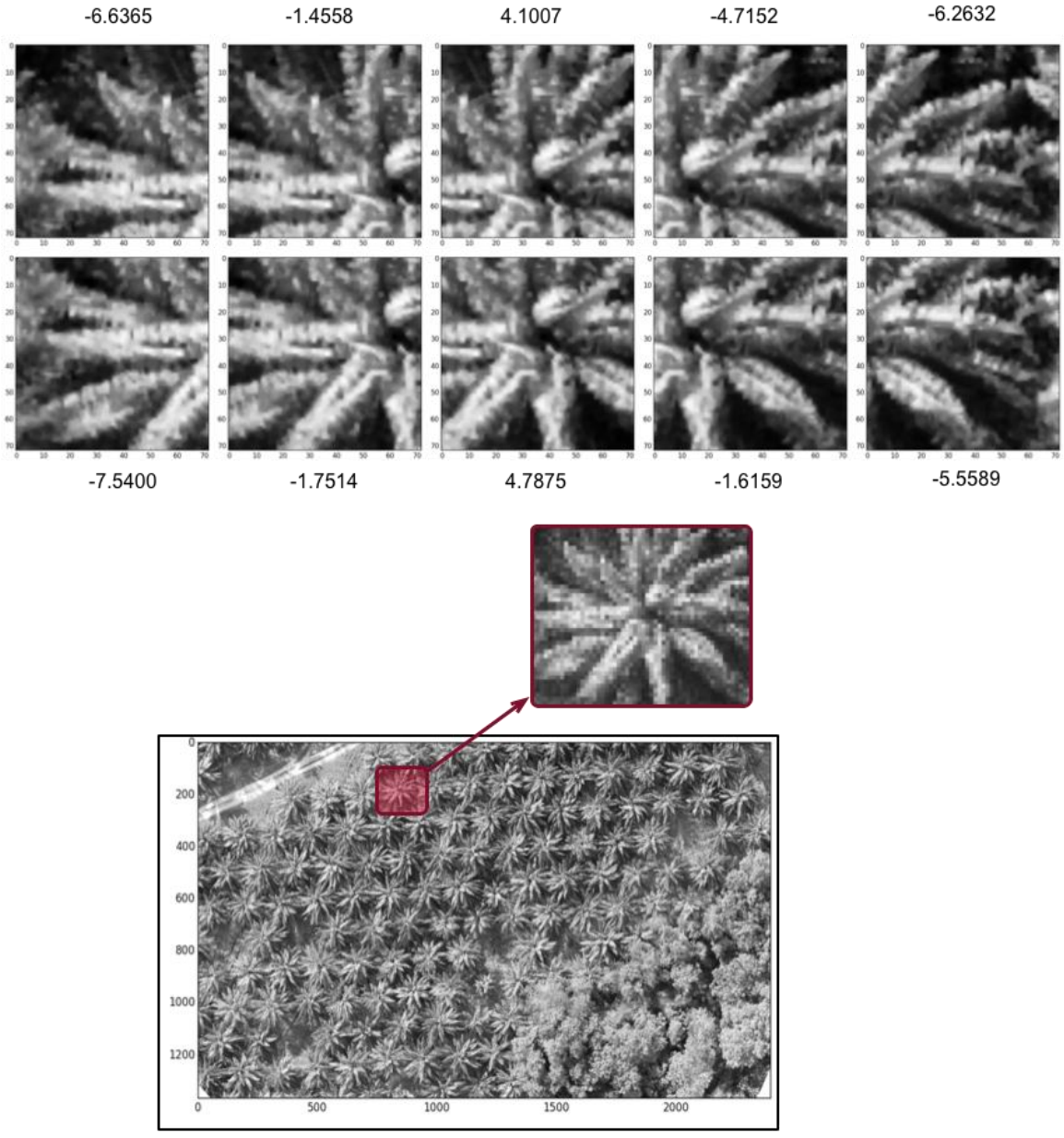


Cuando el nuevo modelo es entrenado, y se utiliza el algoritmo de detección de palmas para evaluar las ventanas de un ortomosaico, a las cuales también se les aplica el filtro, los puntajes que el modelo asigna a las ventanas tienen una mayor diferenciación numérica entre aquellas en donde sí hay palma con respecto a las que no.

La Figura 90 muestra los nuevos scores dados a las mismas ventanas evaluadas en la Figura 72 con el modelo sin filtro. El modelo inicial sólo asignaba un valor

positivo a una de las ventanas donde se sitúa un centro de palma y los valores eran todos más cercanos entre sí, mientras que el nuevo modelo genera puntajes más diferenciados y le asigna valores positivos a las dos ventanas donde se centra la palma.

Figura 90. Puntajes dados por el modelo final para ventanas deslizantes en un área definida



La imagen obtenida como resultado para el ortomosaico de ejemplo cuando es evaluado por el modelo final es presentada en la Figura 91 y el archivo visto desde

el explorador se muestra en la Figura 92. Así mismo el reporte generado en el archivo de texto para esta imagen usando el modelo final se encuentra en la Figura 93.

Figura 91. Imagen de ejemplo con detecciones obtenidas a partir del modelo final

Figura 92. Vista desde el explorador de la imagen resultado de la detección con el modelo final

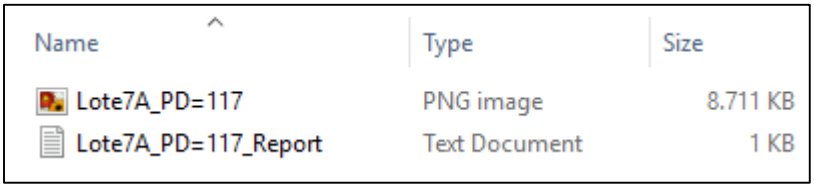
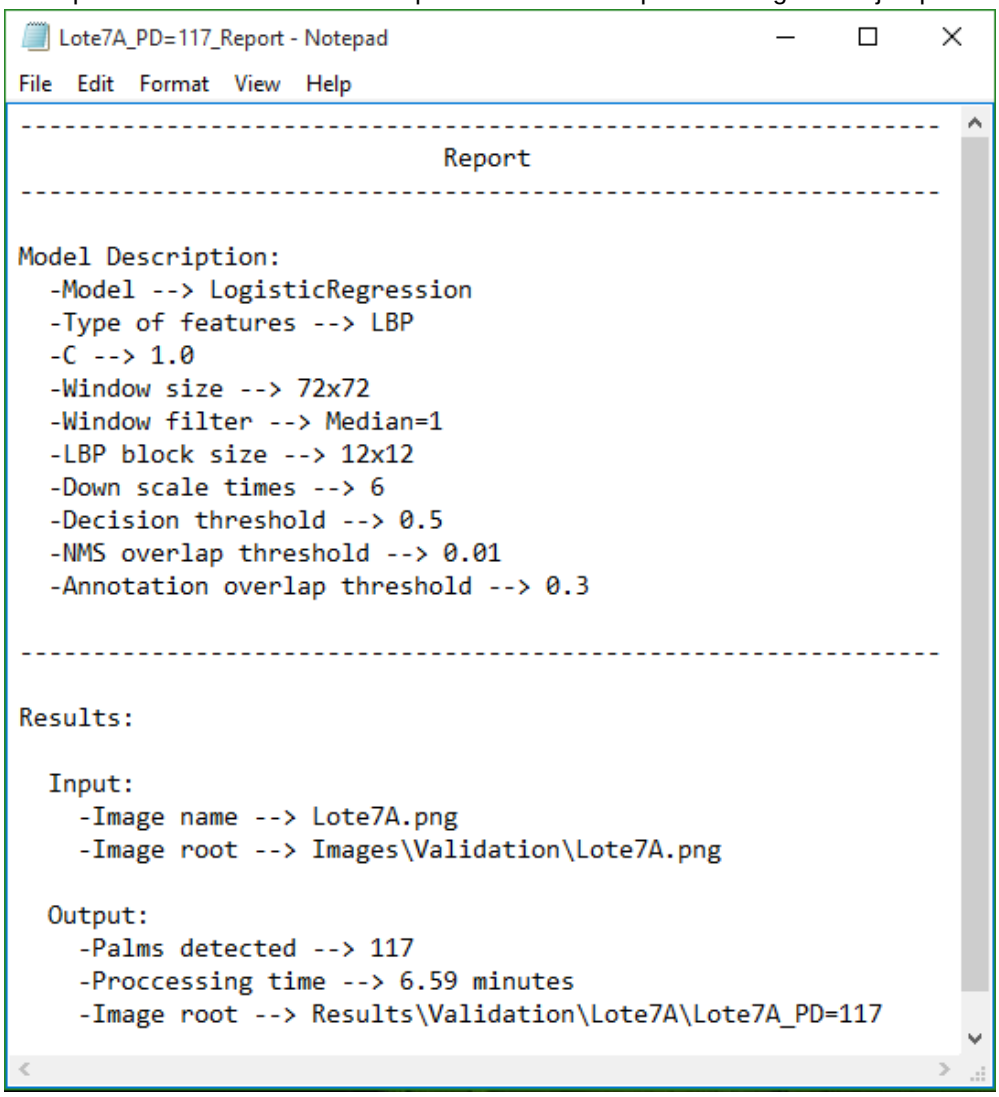


Figura 93. Reporte de detección realizada por el modelo final para la imagen de ejemplo



La Figura 94 compara el comportamiento del modelo inicial con el del modelo final teniendo en cuenta los parámetros de FPPI y tasa de error, mientras que la Figura 95 los compara con respecto a la tasa de detección y a la precisión. Ambas gráficas demuestran una mejora significativa del modelo final con respecto al inicial para los distintos aspectos evaluados en todos los puntos de operación.

Adicionalmente, las gráficas permiten elegir un punto de operación que asegure el máximo rendimiento, el cual corresponde al punto de la curva más cercano al punto ideal y al que le corresponde un umbral de decisión específico. En este caso, el análisis de las gráficas y de la información detallada de cada curva, permite definir un umbral de decisión = 0.5 como punto de operación óptimo.

Figura 94. Comparación de desempeño del modelo final con respecto al inicial: *FPPI vs Miss Rate*

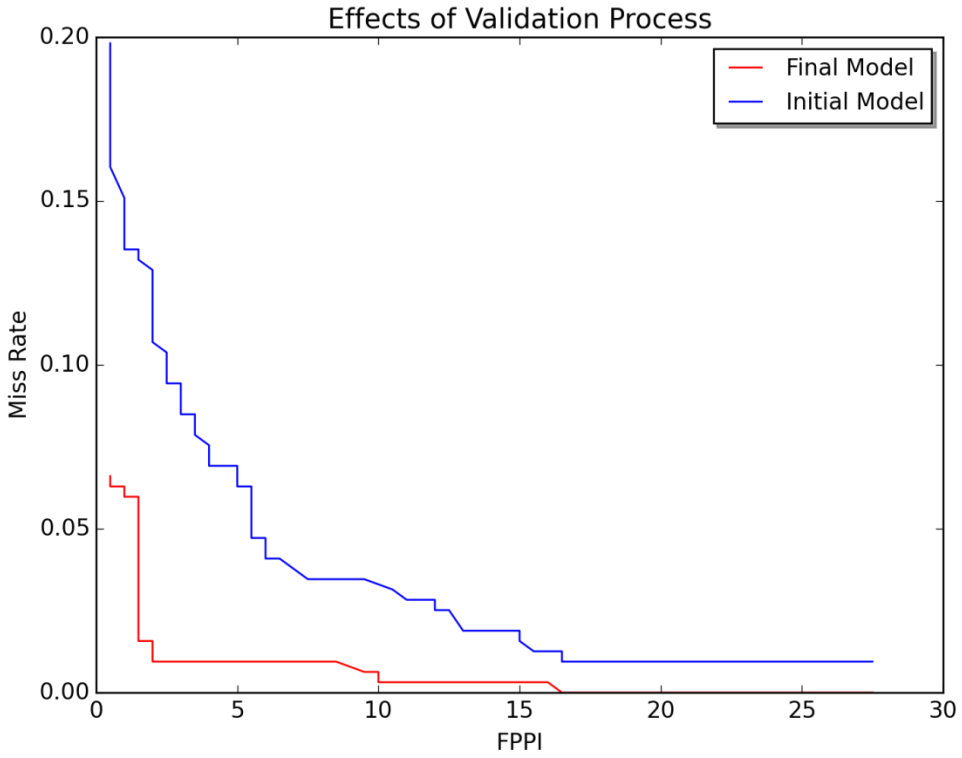
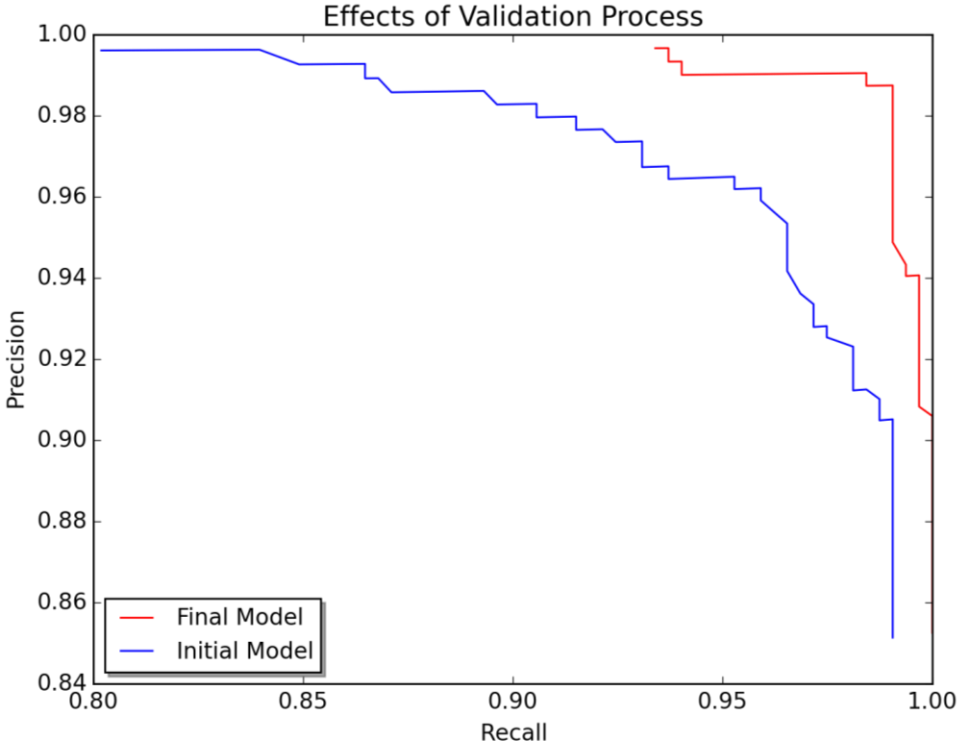


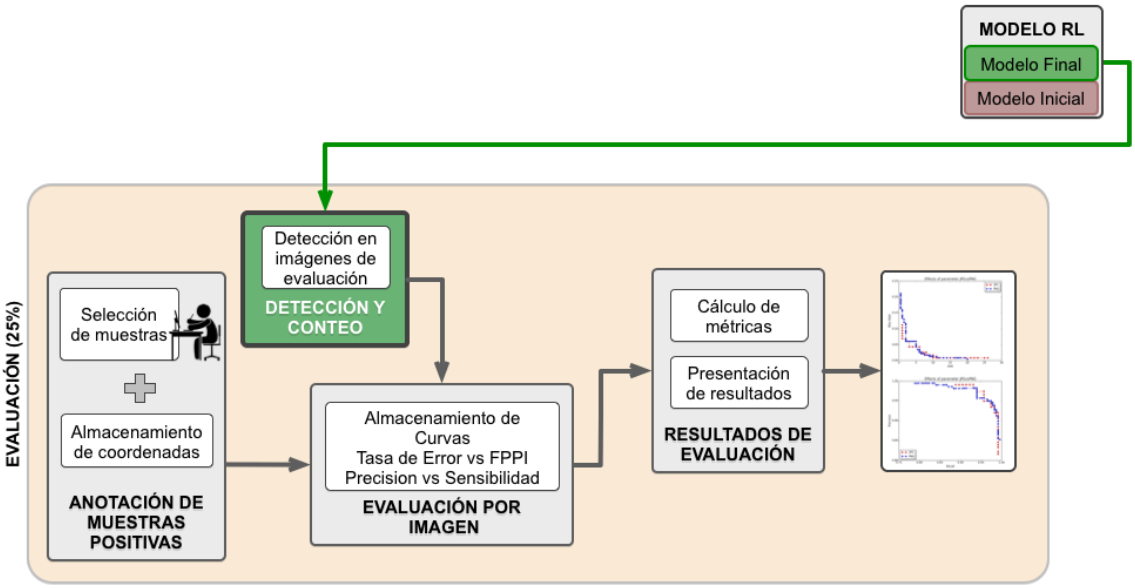
Figura 95. Comparación de desempeño del modelo final con respecto al inicial: *Recall vs Precision*



9.5 EVALUACIÓN DEL SISTEMA

La etapa de validación permite establecer un modelo final cuyos parámetros se han ajustado para obtener el mejor desempeño en un grupo de imágenes que le eran desconocidas durante el proceso de entrenamiento. Sin embargo, al haber optimizado el modelo para ese grupo de ortomosaicos, no se considera correcta la calificación del mismo a partir de ese desempeño. La fase de evaluación del sistema mide el comportamiento del modelo final para nuevos ortomosaicos sin realizar ningún ajuste previo a su utilización. La Figura 96 presenta el diagrama de bloques con las tareas que componen esta fase.

Figura 96. Diagrama de bloques de la fase de evaluación del sistema



9.5.1 Anotación de muestras positivas

Al igual que en la fase de validación, se requieren las ubicaciones reales de las palmas en los ortomosaicos a evaluar para compararlos con los obtenidos por el programa de detección. Este proceso es realizado por una persona utilizando el procedimiento descrito en la sección 9.2.1 Anotación de muestras positivas y negativas en la página 75, pero limitándose sólo a las muestras positivas.

9.5.2 Evaluación por imagen

Para analizar el desempeño del modelo es necesario realizar el mismo proceso de evaluación por imagen descrito en la sección 9.4.2 Evaluación por imagen en la página 100.

El conjunto de evaluación se compone de dos ortomosaicos. El primero de ellos incluye sólo palmas adultas como las que se encuentran en la mayor parte de la base de datos de entrenamiento y en la totalidad del conjunto de validación, las cuales son la prioridad de estudio del proyecto. El segundo ortomosaico presenta sólo palmas pequeñas, que representan un desafío al modelo y cuya detección es un valor agregado. Debido a la diferencia contextual de las imágenes, cada una es analizada y evaluada de forma independiente.

1. EVALUACIÓN EN PALMAS GRANDES

   Al implementar la evaluación por imagen en el primer ortomosaico, se obtiene la curva de desempeño del modelo con respecto a los parámetros de FPPI y tasa de error presentada en la Figura 97 y la gráfica de rendimiento con respecto a la tasa de detección y precisión mostrada en la Figura 98.

Figura 97. Evaluación por imagen en palmas grandes: *FPPI vs Miss Rate*

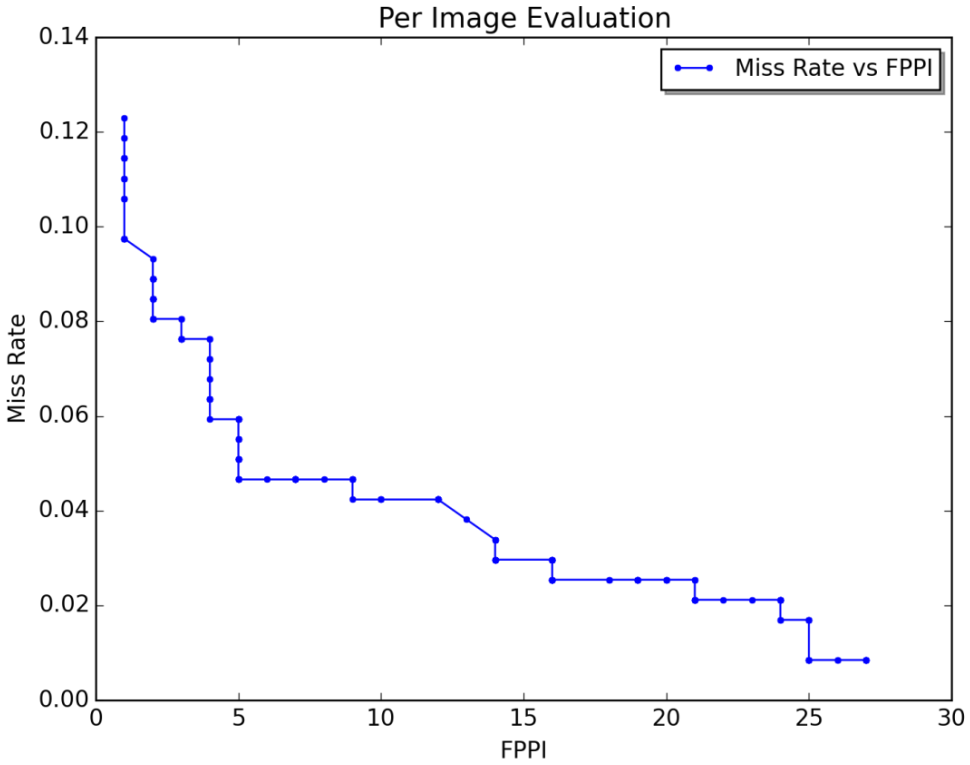
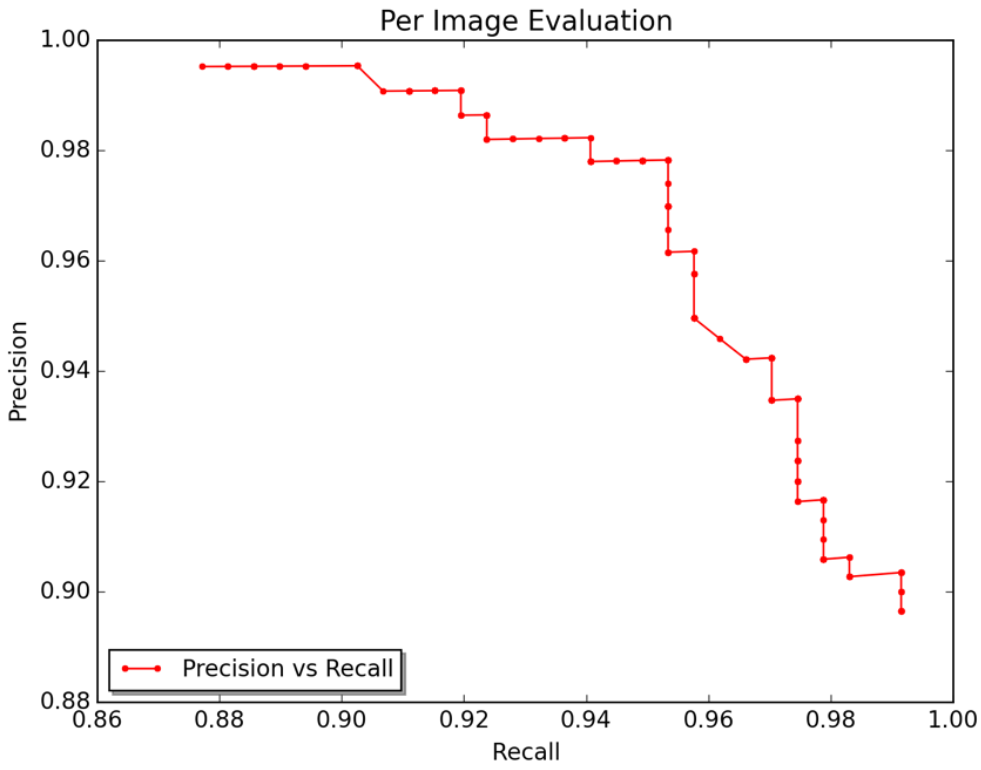


Figura 98. Evaluación por imagen en palmas grandes: *Recall vs Precision*



2. EVALUACIÓN EN PALMAS PEQUEÑAS

   Como resultado de la evaluación por imagen en el segundo ortomosaico, se obtiene la curva de desempeño del modelo con respecto a los parámetros de FPPI y tasa de error presentada en la Figura 99 y la gráfica de rendimiento con respecto a la tasa de detección y precisión mostrada en la Figura 100.

   Como se esperaba, los niveles de la tasa de error que presenta el modelo para las palmas pequeñas es mucho mayor que para las grandes, y por tanto la tasa de detección disminuye de forma proporcional. Sin embargo el sistema responde a palmas pequeñas con una alta precisión, lo que indica que a pesar de haber una baja tasa de detección de verdaderos positivos, el sistema no produce grandes cantidades de falsos positivos.

Figura 99. Evaluación por imagen en palmas pequeñas: *FPPI vs Miss Rate*

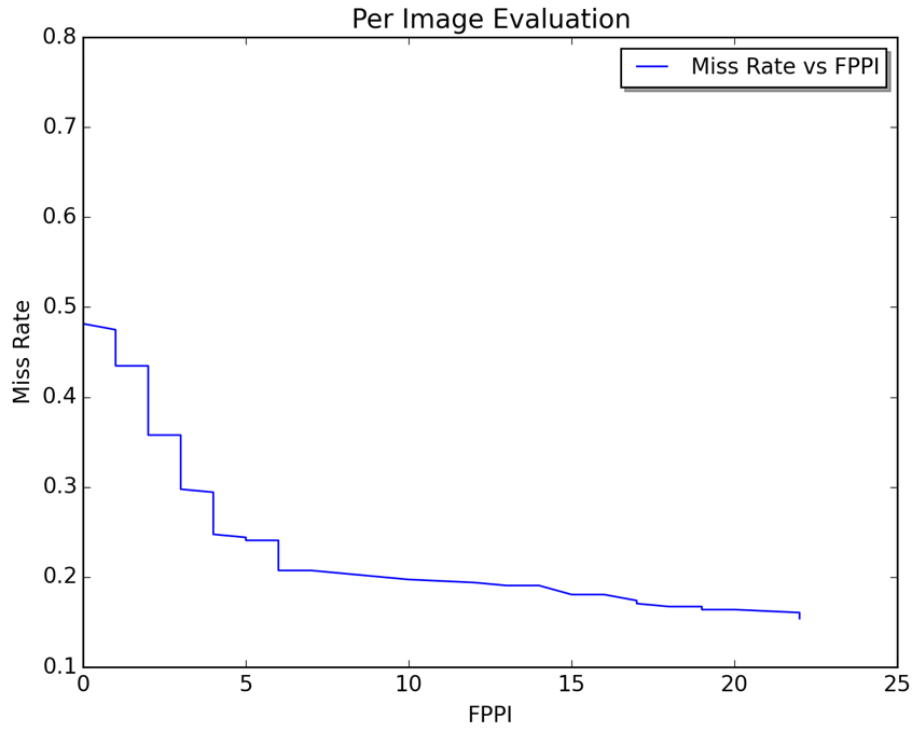
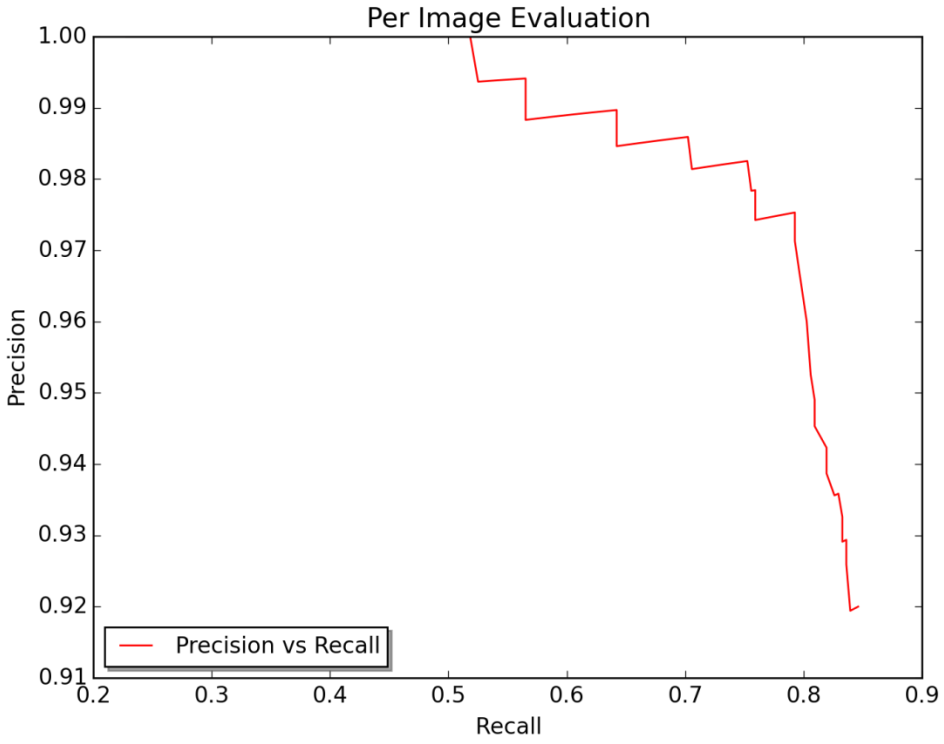


Figura 100. Evaluación por imagen en palmas pequeñas: *Recall vs Precision*



9.5.3 Resultados de la evaluación

1. RESULTADOS EN PALMAS GRANDES
   La fase de validación definió el uso de un umbral de decisión = 0.5, el cual coincide con el punto de operación óptimo en las curvas de evaluación por imagen del primer ortomosaico.

   Al utilizar el programa de detección con esta condición sobre el ortomosaico de palmas grandes, se identifican 230 palmas que se ven reflejadas en la imagen obtenida presentada en la Figura 101 y se genera un reporte que se almacena en un archivo de texto, como se presenta en la Figura 102. Este contiene las métricas de rendimiento del sistema para el modelo final.

2. RESULTADOS EN PALMAS PEQUEÑAS
   Cuando se utiliza el programa de detección en palmas pequeñas con el umbral definido por el grupo de validación las detecciones obtenidas se observan en la Figura 103, en donde se aprecia la disminución de la tasa de detección logrando contar solo 132 objetos.

   Es por esto que se analiza la evaluación por imagen y se define que las palmas pequeñas requieren un umbral mucho más bajo en el modelo actual. Se prueba de nuevo la detección en el ortomosaico con un umbral de decisión = -1.5, para el cual se obtienen 243 detecciones dibujadas en la Figura 104 y el reporte de evaluación de la Figura 105.

Figura 101. Ortomosaico evaluado para palmas grandes con las detecciones obtenidas por el modelo final

Figura 102. Reporte de evaluación para palmas grandes

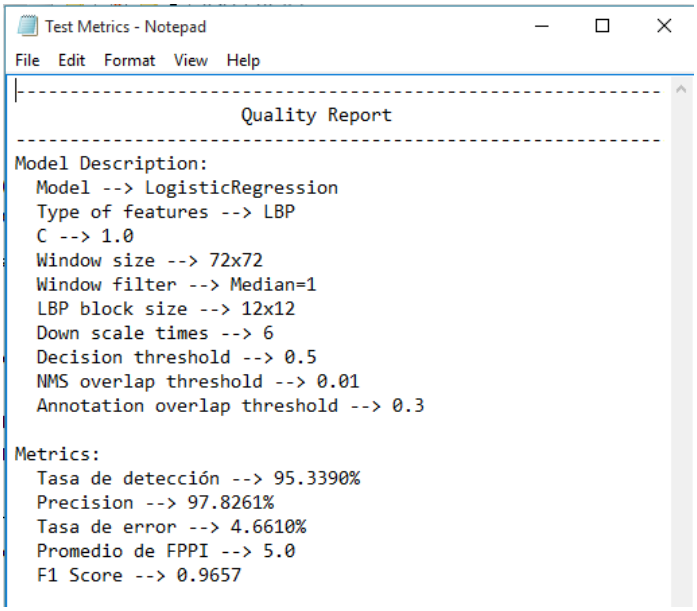


Figura 103. Ortomosaico de evaluación de palmas pequeñas con detecciones usando el umbral definido para palmas grandes

Figura 104. Ortomosaico de evaluación de palmas pequeñas con detecciones usando nuevo umbral específico para palmas pequeñas

Figura 105. Reporte de evaluación para palmas pequeñas

Test Metrics - Notepad

File Edit Format View Help

```
------------------------------------------------------------
                    Quality Report
------------------------------------------------------------
Model Description:
  Model --> LogisticRegression
  Type of features --> LBP
  C --> 1.0
  Window size --> 72x72
  Window filter --> Median=1
  LBP block size --> 12x12
  Down scale times --> 6
  Decision threshold --> -1.5
  NMS overlap threshold --> 0.01
  Annotation overlap threshold --> 0.3

Metrics:
  Tasa de detección --> 79.2642%
  Precision --> 97.5309%
  Tasa de error --> 20.7358%
  Promedio de FPPI --> 6.0
  F1 Score --> 0.8745
```

Como resultado de la evaluación se observa que con el modelo final trabajando en el punto de operación de 0.5, el sistema consigue para las palmas grandes, una tasa de detección del 95.34% con un error respectivo del 4.66%, lo que indica que es capaz de identificar la mayoría de los resultados relevantes. La precisión del 97,83% indica que el sistema es capaz de retornar sustancialmente más resultados relevantes que irrelevantes, de forma que son muchas más las detecciones verdaderas que los falsos positivos, los cuales se sitúan en un promedio de 5 falsos positivos por imagen.

Los ortomosaicos obtenidos cubren zonas de interés con un promedio de 150 a 250 palmas, de forma que la tasa de error del sistema causa que aproximadamente entre 7 a 12 palmas pueden no llegar a ser identificadas. En el cálculo del conteo total de palmas, el promedio de falsos positivos por imagen contrarresta algunas de estas palmas no detectadas.

En el caso de las palmas pequeñas, el sistema no ofrece buenos resultados si se maneja el mismo punto de operación usado para las palmas grandes. Por esta razón se plantea el uso de un umbral distinto situado en -1.2, lo que demuestra que la textura de estas palmas se ubica hacia la zona negativa del hiperplano de frontera.

La evaluación con este tipo de plantas genera una tasa de detección del 79.26% con un error respectivo del 20.74%, disminuyendo notablemente la capacidad de identificar en su totalidad las palmas con respecto a la situación de las palmas grandes. La alta precisión del 97.53% indica que, aunque no se identifican todos los especímenes deseados, las detecciones realizadas sí son en su mayoría palmas y no falsos positivos, cuyo promedio por imagen se sitúa en 6 plantas.

Finalmente, ante la medida de exactitud score F1, que considera tanto la tasa de detección como la precisión, el sistema obtiene una calificación de 0.97 para las palmas grandes y de 0.87 para las palmas pequeñas, siendo coherente con el comportamiento dado en cada situación.

Para visualizar el rendimiento del modelo final, se aplica el programa de detección y conteo de palmas en todos los ortomosaicos adquiridos durante el trabajo de campo, obteniendo los resultados presentados en la Figura 106. En ella se puede observar que las detecciones para los ortomosaicos de zonas con palmas grandes obtienen un error que se enmarca dentro del 5% determinado, mientras que para las palmas pequeñas la detección no es tan eficiente.

Adicionalmente, se presenta el número total de ventanas que son analizadas como candidatas para cada imagen. La Figura 107 detalla la cantidad de ventanas estudiadas en cada nivel de la pirámide.

Por último, la Figura 108, la Figura 109, la
Figura 110, la Figura 111, la Figura 112, la Figura 113, la Figura 114 y la Figura 115 presentan las imágenes con las detecciones obtenidas para cada ortomosaico.

Figura 106. Resultados de pruebas del modelo final en todos los ortomosaicos

RESULTADOS DE PRUEBAS EN ORTOMOSAICOS
(PALMAS GRANDES)

Imagen		Palmas detectadas	Palmas reales	Ventanas analizadas
ENTRENAMIENTO				
	El Portico 2	236	250	33311
	Lote 3B	179	190	30073
	Lote 7B	183	183	34057
VALIDACIÓN				
	Lote 6	203	203	21195
	Lote 7A	117	118	15722
EVALUACIÓN				
	El Portico 1	230	237	11131
	TOTALES	**1148**	**1181**	**145489**

RESULTADOS DE PRUEBAS EN ORTOMOSAICOS
(PALMAS PEQUEÑAS)

Imagen		Palmas detectadas	Palmas reales	Ventanas analizadas
ENTRENAMIENTO				
	Lote 5	105	227	18353
	El Portico 3	243	280	30244
	TOTALES	**348**	**507**	**48597**

Figura 107. Dimensiones de los niveles de la pirámide y número de ventanas analizadas para cada ortomosaico

Nivel en pirámide		Original (Nivel 0)			Nivel 1			Nivel 2			Nivel 3		
Imagen		Alto [pixeles]	Ancho [pixeles]	Ventanas Nivel 0	Alto [pixeles]	Ancho [pixeles]	Ventanas Nivel 1	Alto [pixeles]	Ancho [pixeles]	Ventanas Nivel 2	Alto [pixeles]	Ancho [pixeles]	Ventanas Nivel 3
ENTRENAMIENTO													
	El Portico 2	3504	1947	**11520**	3186	1770	**9432**	2897	1610	**7854**	2634	1464	**6372**
	Lote 3B	2767	2227	**10374**	2516	2025	**8549**	2288	1841	**7050**	2080	1674	**5780**
	Lote 5	1190	3212	**6336**	1082	2920	**5280**	984	2655	**4251**	895	2414	**3564**
	Lote 7B	1650	4250	**11792**	1500	3864	**9699**	1364	3513	**7975**	1240	3194	**6600**
VALIDACIÓN													
	Lote 6	1558	2805	**7245**	1417	2550	**6090**	1289	2319	**4940**	1172	2109	**4042**
	Lote 7A	1365	2395	**5390**	1241	2178	**4450**	1129	1980	**3726**	1027	1800	**2993**
EVALUACIÓN													
	El Portico 1	2199	1061	**3870**	2000	965	**3198**	1819	878	**2590**	1654	799	**2144**
	El Portico 3	3061	2022	**10458**	2783	1839	**8550**	2530	1672	**7072**	2300	1520	**5828**

Nivel en pirámide		Nivel 3			Nivel 4			Nivel 5			TOTAL VENTANAS
Imagen		Alto [pixeles]	Ancho [pixeles]	Ventanas Nivel 3	Alto [pixeles]	Ancho [pixeles]	Ventanas Nivel 4	Alto [pixeles]	Ancho [pixeles]	Ventanas Nivel 5	
ENTRENAMIENTO											
	El Portico 2	2634	1464	**6372**	2395	1331	**5292**	2178	1210	**4361**	**33311**
	Lote 3B	2080	1674	**5780**	1891	1522	**4774**	1720	1384	**3920**	**30073**
	Lote 5	895	2414	**3564**	814	2195	**2880**	740	1996	**2378**	**18353**
	Lote 7B	1240	3194	**6600**	1128	2904	**5355**	1026	2640	**4428**	**34057**
VALIDACIÓN											
	Lote 6	1172	2109	**4042**	1066	1918	**3354**	970	1744	**2769**	**21195**
	Lote 7A	1027	1800	**2993**	934	1637	**2479**	850	1489	**2074**	**15722**
EVALUACIÓN											
	El Portico 1	1654	799	**2144**	1504	727	**1769**	1368	661	**1430**	**11131**
	El Portico 3	2300	1520	**5828**	2091	1382	**4816**	1901	1257	**3978**	**30244**
										TOTAL	**194086**

Figura 108. Detecciones obtenidas con el modelo final para el ortomosaico "El Portico2.png"

Figura 109. Detecciones obtenidas con el modelo final para el ortomosaico "Lote 3B.png"

Figura 110. Detecciones obtenidas con el modelo final para el ortomosaico "Lote 5.png"

Figura 111. Detecciones obtenidas con el modelo final para el ortomosaico "Lote 7B.png"

Figura 112. Detecciones obtenidas con el modelo final para el ortomosaico "Lote 6.png"

Figura 113. Detecciones obtenidas con el modelo final para el ortomosaico "Lote 7A.png"

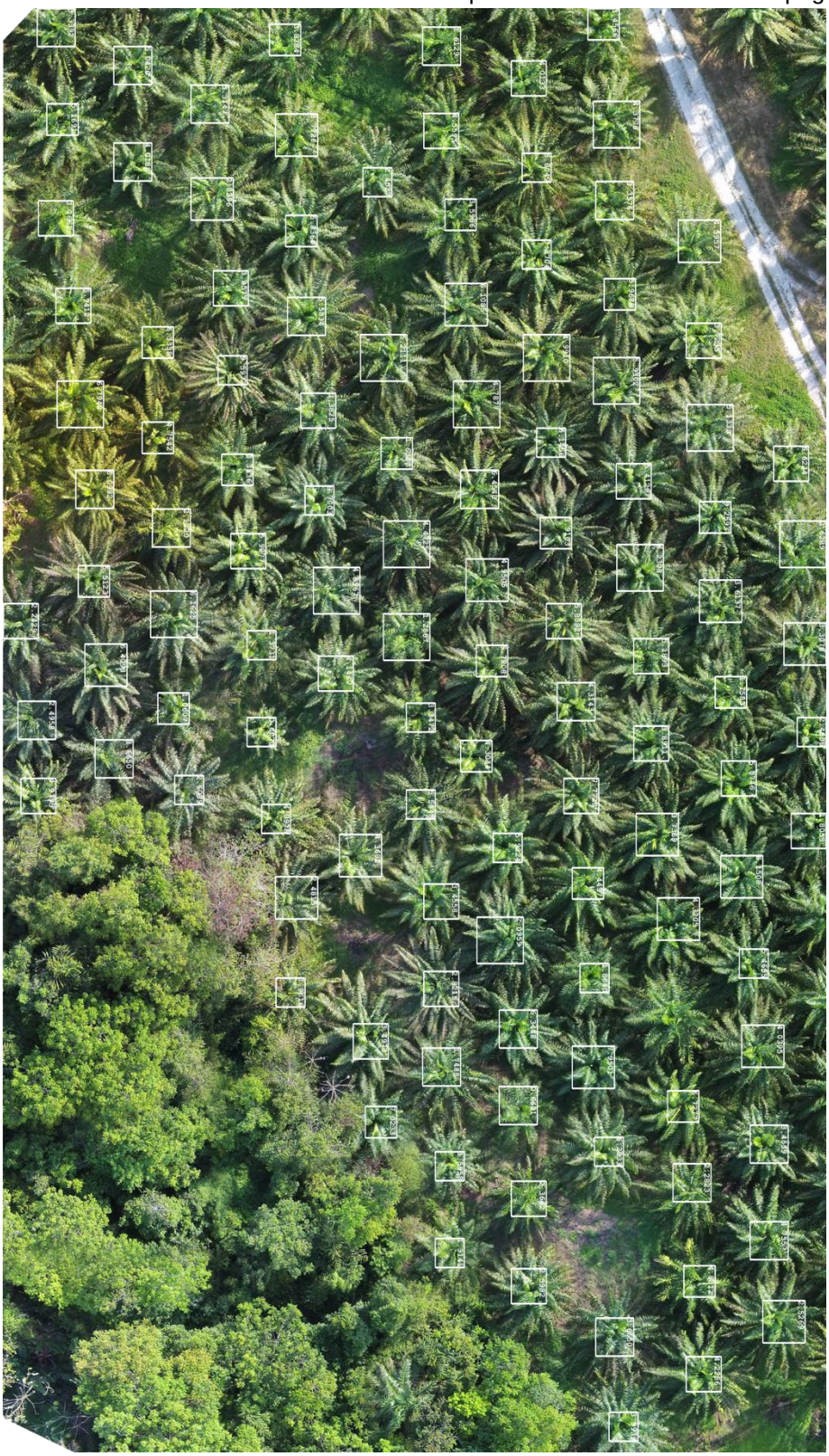


Figura 114. Detecciones obtenidas con el modelo final para el ortomosaico "El Portico1.png"

Figura 115. Detecciones obtenidas con el modelo final para el ortomosaico "El Portico3.png"

10. CONCLUSIONES

En este proyecto se realizó un análisis de los vehículos aéreos no tripulados disponibles en el mercado y se eligió el Phantom2 Vision+ debido a que sus especificaciones técnicas eran las más adecuadas para suplir las necesidades del proyecto, lo que permitió la adquisición de las imágenes requeridas en las plantaciones.

Se utilizó la aplicación Pix4Dmapper Capture como herramienta para diseñar las rutas de vuelo del UAV con el fin de obtener las fotografías adecuadas de la plantación para la elaboración de ortomosaicos, los cuales fueron creados usando el software de fotogrametría Pix4Dmapper y permitieron obtener vistas aéreas de alta calidad solucionando el problema de distorsión de perspectiva presente en las imágenes adquiridas por la aeronave.

Para el desarrollo del software se aplicaron técnicas de aprendizaje automático y procesamiento de imágenes que permitieron obtener un modelo de regresión logística capaz de detectar las palmas a partir de características LBP-Uniforme (*Local Binary Pattern*) apoyado por técnicas de ventana deslizante, escalado piramidal y non-maximum suppression para mejorar la calidad de la decisión.

Inicialmente el objetivo fue realizar censos en las zonas de la plantación con palmas grandes, en donde es imposible visualizar todas desde un punto en tierra. El sistema fue probado directamente con imágenes de plantaciones reales lo que permite evaluar su verdadero desempeño. El error obtenido fue de 4.66% con una exactitud medida por el score F1 de 0.97.

Adicionalmente, se realizaron pruebas sobre zonas con palmas pequeñas, en donde se genera un error del 20.74% con una exactitud medida por el score F1 de 0.87, concluyendo que el sistema sí es capaz de identificarlas, aunque con un error más alto, a pesar de no haber sido optimizado para ellas.

Se demostró que los vehículos aéreos no tripulados son una herramienta exitosa en el monitoreo de plantaciones pues permiten la obtención de información útil de forma más económica que los satélites y con la ventaja de no presentar problemas de nubosidad. Adicionalmente, con el UAV se realiza la labor de recorrer la plantación evitando que el trabajador deba desplazarse a través de la misma, lo que facilita las condiciones de trabajo y permite que esta tarea pueda ser actualizada con mayor frecuencia.

Además del sistema de detección y conteo desarrollado, los ortomosaicos obtenidos resultan ser una herramienta muy útil para los palmicultores ya que les permiten realizar labores de monitoreo sobre las áreas mapeadas, brindándoles la posibilidad de obtener información que hasta el momento desconocían en algunas zonas, y que es captada por expertos con la sola visualización de los mapas.

De esta forma, se logró el desarrollo de un sistema de conteo en plantaciones de palma de aceite que permite adquirir las imágenes de una zona de interés, crear un ortomosaico de la misma y detectar automáticamente el número de palmas contenidas.

11. RECOMENDACIONES Y DIRECCIONES FUTURAS

Con el propósito de continuar con el trabajo realizado y como propuesta para proyectos futuros en la línea de procesamiento de imágenes, aprendizaje automático y uso de aeronaves, se plantean las siguientes recomendaciones con respecto al sistema de conteo:

- Utilizar un UAV con mayor rango de vuelo que permita abarcar grandes distancias con el fin de obtener ortomosaicos de lotes completos haciendo que la información obtenida sea de mayor utilidad para los palmicultores.

- Desarrollar un modelo capaz de identificar indicadores de enfermedades y plagas basándose en fotografías aéreas de las palmas.

- Adaptar diferentes tipos de cámaras multiespectrales que le permitan al UAV recopilar datos desde otras bandas del espectro de luz que pueden brindar información importante acerca del estado fitosanitario de las palmas.

- Desarrollar una aplicación que presente todos los lotes de una plantación en forma de mapas georreferenciados en donde se incluya el censo actualizado por zonas y la ubicación de palmas enfermas.

BIBLIOGRAFÍA

ANWER, Rao; VÁSQUEZ, David y LÓPEZ, Antonio. Color contribution to part-based person detection in different types of scenarios. En: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2011. vol.6855 LNCS, no.2.

ARAQUE, Leonardo y FORERO, Diana. Análisis de los sistemas de captura y procesamiento de información para la toma de decisiones en el manejo de los insectos defoliadores de la palma de aceite en Colombia. En: Palmas. 2009. vol 30, no. 1.

ASPINALL, Richard. Use of logistic regression for validation of maps of the spatial distribution of vegetation species derived from high spatial resolution hyperspectral remotely sensed data. En: Ecological Modelling. 2002. vol.157, no.2-3.

BARRIENTOS, A, *et al*. Vehículos aéreos no tripulados para uso civil. Tecnología y Aplicaciones. Grupo de Robótica y Cibernética. Universidad Politécnica de Madrid.

BÉJAR, Javier. Aprendizaje automático. En: Diapositivas de curso (2014/2015: Barcelona, España). Inteligencia Artificial. Facultad de Sistemas Informáticos. Universitat Politècnica de Catalunya.

BENSAEED, O, *et al.* Oil palm fruit gradient using a hyperspectral device and machine learning algorithm. En: IOP Conference Series: Earth and Environmental Science. 2014. vol.20, no.1.

BERNI, J, *et al.* Thermal and Narrowband Multispectral Remote Sensing for Vegetation Monitoring From an Unmanned Aerial Vehicle. En: IEEE Transactions on Geoscience and Remote Sensing. Marzo, 2009. vol.47, no.3.

BISHOP, Christopher. Pattern Recognition and Machine Learning. Singapore: Springer, 2006. ISBN-10: 0-387-31073-8.

BRADSKI, Gary y KAEHLER, Adrian. Learning OpenCV. Computer Vision with the OpenCV Library. Sebastopol: O'Reilly Media, Inc., 2008. p. 2.

COSTA, F.G, *et al.* The use of unmanned aerial vehicles and wireless sensor network in agricultural applications. En: Geoscience and Remote Sensing Symposium (IGARSS) (22-27, julio, 2012: Munich). IEEE International, 2012.

DADWAL, M. y BANGA, V. Estimate ripeness level of fruits using RGB color space and fuzzy logic technique. En: International journal of Engineering and Advanced Technology. vol.2, no.1.

DONG, Jia-Jyun, *et al.* Logistic regression model for predicting the failure probability of a landslide dam. En: Engineering Geology. Octubre, 2011. vol.117, no.1-2.

FADILAH, Norasyikin, *et al.* Intelligent color vision system for ripeness classification of oil palm fresh fruit bunch. En: Sensors. 2012. vol.12, no.10.

FEDEPALMA. Informe de Gestión 2012. [online]. Disponible en Fedepalma < http://issuu.com/fedepalma/docs/informe_de_gestio__n_fedepalma_2012>

GOMEZ, Sergio y SEGURA, Fredy. An aerial monitoring system (AMS) for detecting bud-rot diseases on oil palms. Tesis de Maestría. Bogotá D.C.: Universidad de los Andes.

GONZÁLEZ, Rafael C. y WOODS, Richard E. Digital Image Processing. 3 ed.New Jersey: Prentice Hall, Inc., 2008.

GUERRERO, Edgar y BENAVIDES, Gustavo. Automated system for classifying Hass avocados based on image processing techniques. En: IEEE Colombian Conference on Communications and Computing (COLCOM) (4-6, junio, 2014: Bogotá D.C., Colombia). Memorias. Bogotá D.C.: IEEE, 2014.

HOYOS, Michel y RINCÓN, Víctor. Uso de dispositivos móviles para la captura de datos en campo con formularios electrónicos a través del programa Cybertracker. En: Palmas. 2014. vol 35, no. 4.

JAFFAR, A, *et al.* Photogrammetric grading of oil palm fresh fruit bunches. En: International Journal of Mechanical and Mechatronics Engineering. Diciembre, 2009. vol.9, no.10.

KWESI, Nooni. Oil palm mapping using Support Vector Machine with LANDSAT ETM+ data. Tesis de Maestría. Master of Science in Geo- Information Science and Earth Observation. Enschede, The Netherlands: Faculty of Geo-Information

Science and Earth Observation of the University of Twente and the Faculty of Renewable Natural Resources of the Kwame Nkrumah University of Science & Technology, 2012.

LÓPEZ, Antonio; VALVENY, Ernest y VANRELL, María. Detección de Objetos [Curso online]. Universitat Autònoma de Barcelona: mayo-julio, 2015. Disponible en: Coursera https://www.coursera.org/course/deteccionobjetos

MARIAU, Dominique. Vigilancia sanitaria de las plantaciones de palma de aceite y cocotero. En: Palmas. 1995. vol 16, no. 1.

MAY, Z. y AMARAN, M. Automated ripeness assessment of oil palm fruit sing RGN and fuzzy logic technique. En: WSEAS International Conference on Mathematical and Computational Methods in Science and Engineering (MACMESE) (13: 2011, Stevens Point, Wisconsin, USA). World Scientific and Engineering Academy and Society (WSEAS), 2011.

MENARD, Scott. Applied logistic regression analysis. 2 ed. Thousand Oaks, CA: Sage University Papers Series on Quantitative Applications in the Social Sciences, 2002.

MNIH, Volodymyr y HINTON, Geoffrey. Learning to detect roads in high-resolution aerial images. En: European Conference on Computer Vision (11: 5-11, septiembre, 2010: Heraklion, Crete, Greece). Proceeding. Berlin: Springer-Verlag, 2010.

NG, Andrew. Model selection and Train/Validation/Test sets. Advice for applying machine learning. Machine learning [Curso online]. Stanford University, 2015. Disponible en: Coursera < https://www.coursera.org/learn/machine-learning/ >

NG, Patrick, *et al*. La teleobservación y las tecnologías digitales para el manejo de las plantaciones. En: Palmas. 2013. vol. 34, no. Especial, Tomo I.

NOBOU, Spike. Application of information extraction in the plant growth based on image processing. En: Kansai Branch of the Agricultural Society of Machinery Engineers. 1992. vol. 72.

PAULRAJ, M, *et al.* Color recognition algorithm using a neural network model in determining the ripeness of a banana. En: International Conference on Man-Machine Systems (ICoMMS) (11-13, octubre, 2009: Batu Ferringhi, Penang, Malasia). Proceedings.

PEDREGOSA, Fabian, *et al.* Scikit-learn: Machine Learning in Python. En: Journal of Machine Learning Research. 2011. vol.12.
PIETIKÄINEN, Matti, *et al.* Computer vision using local binary patterns. Springer, 2011. ISBN:9780857297471.
Pix4D. Pix4D Introduction Guide [Diapositivas]. En: Pix4D Support Site. Octubre, 2013. Disponible en:
<https://support.pix4d.com/hc/en-us/articles/202561499-Pix4D-Introduction-Guide>. Revisado en: Agosto, 2015.

POHL, Christine. Mapping palm oil expansion using SAR to study the impact on the CO_2 cycle. En: IGRSM International Remote Sensing & GIS Conference and Exhibition (7: 21-22, abril, 2014: Kuala Lumpur). IOP Conference Series: Earth and Environmental Science. 2014. vol. 20, no. 1.

Portafolio. Colombia, cuarto productor de aceite de palma en el el mundo. 18, septiembre, 2014. Disponible en < http://www.portafolio.co/especiales/portafolio-21-aniversario/colombia-productor-aceite-palma-2014> Revisado en: Agosto, 2015.

RIZAM, Shah *et al.* Non-destructive watermelon ripeness determination using image processing and artificial neural network (ann). En: International Journal of Electrical and Computer Engineering. vol.4. no.6.

RSPO (Roundtable on Sustainable Palm Oil). Where is palm oil grown? En: Green Palm Sustainability. 2014. Disponible en: < http://greenpalm.org/about-palm-oil/where-is-palm-oil-grown-2>. Revisado en: Agosto, 2015.

RSPO (Roundtable on Sustainable Palm Oil). Why is palm oil important? En: Green Palm Sustainability. 2014. Disponible en: < http://greenpalm.org/about-palm-oil/why-is-palm-oil-important>. Revisado en: Agosto, 2015.

Scikit-learn, Sitio web. Historia, Documentación. Disponible en: <http://scikit-learn.org/stable/about.html>. Revisado en Agosto, 2015.

TESAURO, Gerald. TD-Gammon, a self-teaching backgammon program, achieves master-level play. En: Neural Computation. 1994.

TEUTSCH, Michael. Moving object detection and segmentation for remote aerial video surveillance. Disertación para optar por el grado de Doctor de Ingeniería. Karlsruher Institut für Technologie (KIT). Facultad de Informática, 2014.

TIAN, Y.W y LI, C.H. Research on Recognition of Cucumber Disease Based on Image Processing in Sunlight Greenhouse. En: Journal of Agricultural Mechanization Research. 2006. vol.2.

VAROQUAUX, Gaël, *et al.* Scikit-learn: Machine learning without learning the machinery. En: GetMobile: Computing and Communications. 2015. vol.19, no.1.

WALT, Stéfan van der, *et al.* Scikit-image: Image processing in Python. En: PeerJ 2:e453. Abril, 2014. Disponible en: <http://dx.doi.org/10.7717/peerj.453> Revisado en: Agosto, 2015.

YANG, Lin, *et al.* Tree detection from aerial imagery. En: ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (17: 4-6, noviembre, 2009: Seattle, WA, USA). Proceedings. New York: ACM New York, 2009.

ZHANG, Lu y MCCARTHY, Michael. Measurement and evaluation of tomato maturity using magnetic resonance imaging. En: Postharvest Biology and Technology. Mayo, 2012. vol.67.